

SIEMENS



Cooperates
with Education

Automation

SIEMENS

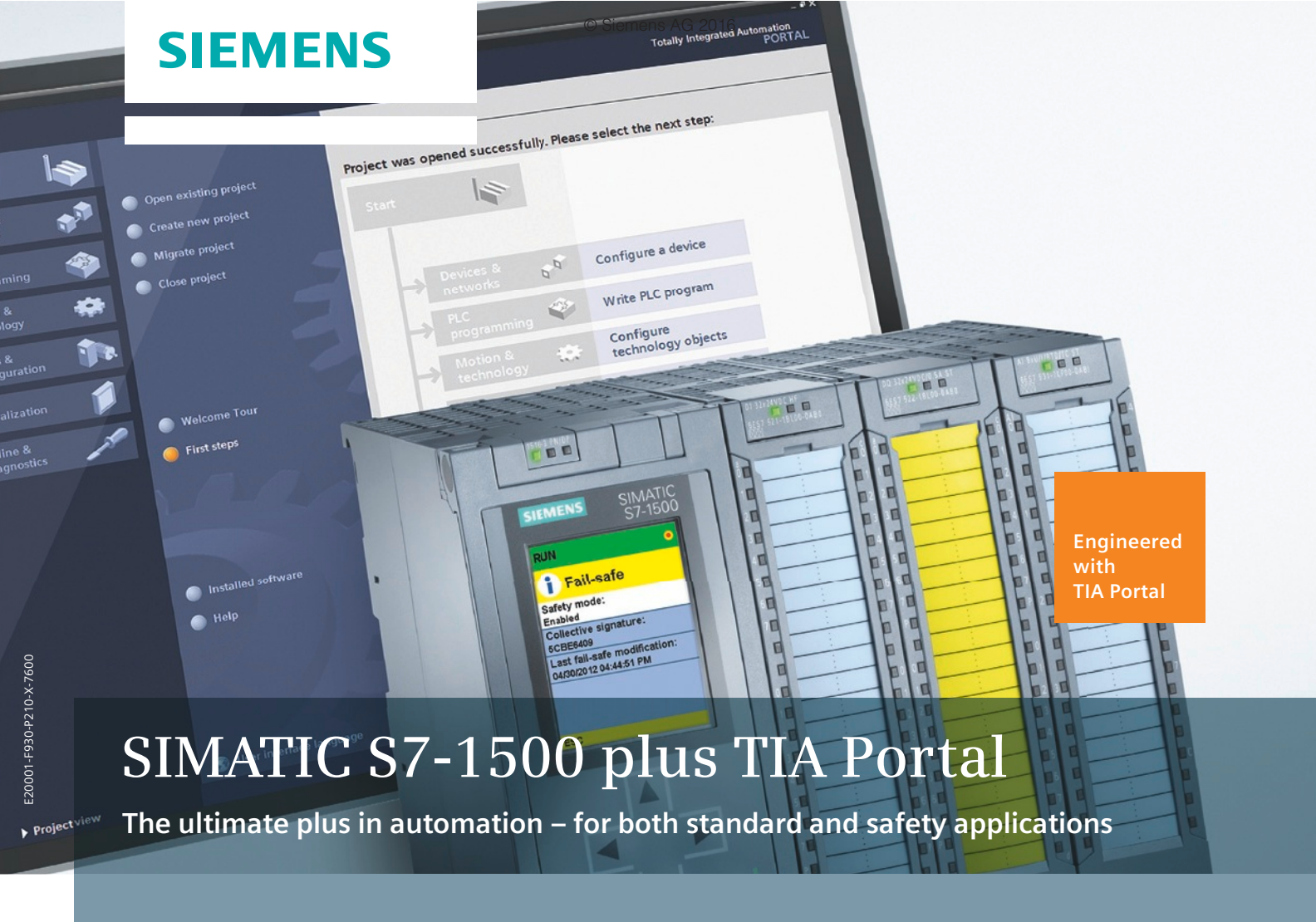
SCE Training Curriculum

Automation System SIMATIC S7-1500

TIA Portal
Module
0XX-600

Edition
04/2016

siemens.com/sce



Engineered
with
TIA Portal

SIMATIC S7-1500 plus TIA Portal

The ultimate plus in automation – for both standard and safety applications

Highest performance – highest usability:
The SIMATIC S7-1500 controller range with the high-end CPU 1518 represents the new generation of controllers in the TIA Portal and a milestone in automation.

Failsafe versions are now also available in the form of the CPUs 1516F and 1518F – for standard and failsafe automation in a single system.

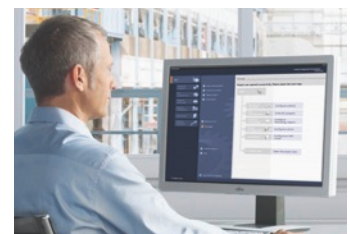
Your plus for power:

- + Outstanding **system performance** for shortest response times and highest quality of control
- + **Technology integrated** for perfect integration of drives through motion control functionalities and PROFIdrive
- + **Security integrated** – consistently incorporated for highest investment protection

Your plus for efficiency:

- + **Innovative design and easy handling** for simple usage and commissioning as well as safe operation
- + **Integrated system diagnostics** for full transparency of the plant status, automatically generated and consistently displayed
- + **TIA Portal** for highest engineering efficiency and reduced project costs

Experience the new controller's highlights online: siemens.com/s7-1500



Intuitive, efficient, proven: Totally Integrated Automation Portal (TIA Portal) redefines engineering.

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Continued training

For regional Siemens SCE continued training, get in touch with your regional SCE contact

siemens.com/sce/contact

Additional information regarding SCE

siemens.com/sce

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational facilities and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems, which means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact person: Roland Scheuerer roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is explicitly not permitted. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	General overview of the modules	4
1.1	Subject planning	4
1.2	Basic modules	5
1.3	Advanced modules	8
1.4	Special modules	10
2	Concept description	12
2.1	Material at a glance	12
2.2	Structure of the individual chapters based on the teaching methodology	13
3	Additional information	14

MODULE AND CONCEPT DESCRIPTION

1 General overview of the modules

The module description is intended to help you to select the contents that are relevant, interesting and technically suited to you.

The numbers of the chapters contain 6 digits, e.g., 000-000 for the module description chapter. The first 2 digits refer to the subject area. The third digit is also the highest outline level of the subject. Most of the time, "1" stands for the S7-1200 controller and "2" for the S7-1500. Where present, "3" stands for the S7-300 controller. The digits after the dash refer to a particular chapter. If this chapter is available for different controllers, there is only a difference in the third digit of the first three digits and not in the last digits.

1.1 Subject planning

The following figure represents an overview of the topics contemplated for SCE according to the concept described here. These topics are very wide-ranging and can therefore only be dealt with individually. For this reason, the topics presented here are further subdivided in the following sections into basic, advanced and future topics (= modules). Basic modules contain SCE Training Curriculums for beginning learners but can also be used for advanced learners. Advanced modules are recommended in particular for advanced learners or individuals who have completed the basic modules. Special modules contain all other topics regarded as interesting and promising for the future. Additions are planned here in any event. Always check the website siemens.com/sce.

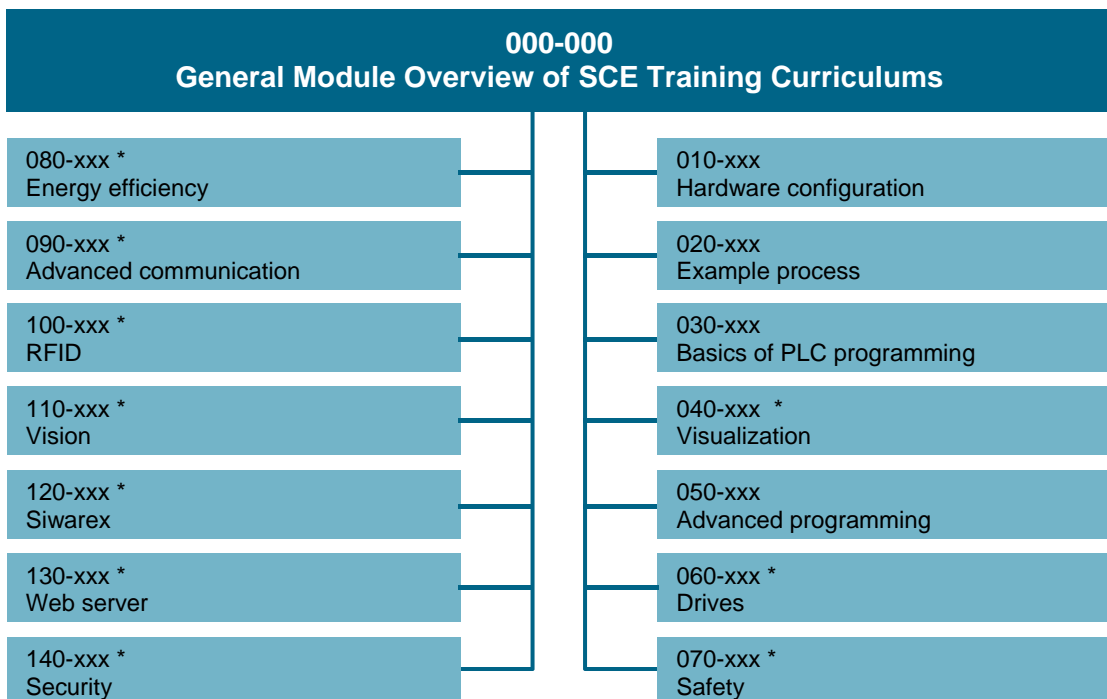


Figure 1: Overview of the available topics

* Upcoming

1.2 Basic modules

Besides the module description that you have before you, the basic modules include the following topics: "Hardware configuration", "Example processes" and "Basics of PLC programming" (see Figure 2). The structure of these modules is explained in the following.

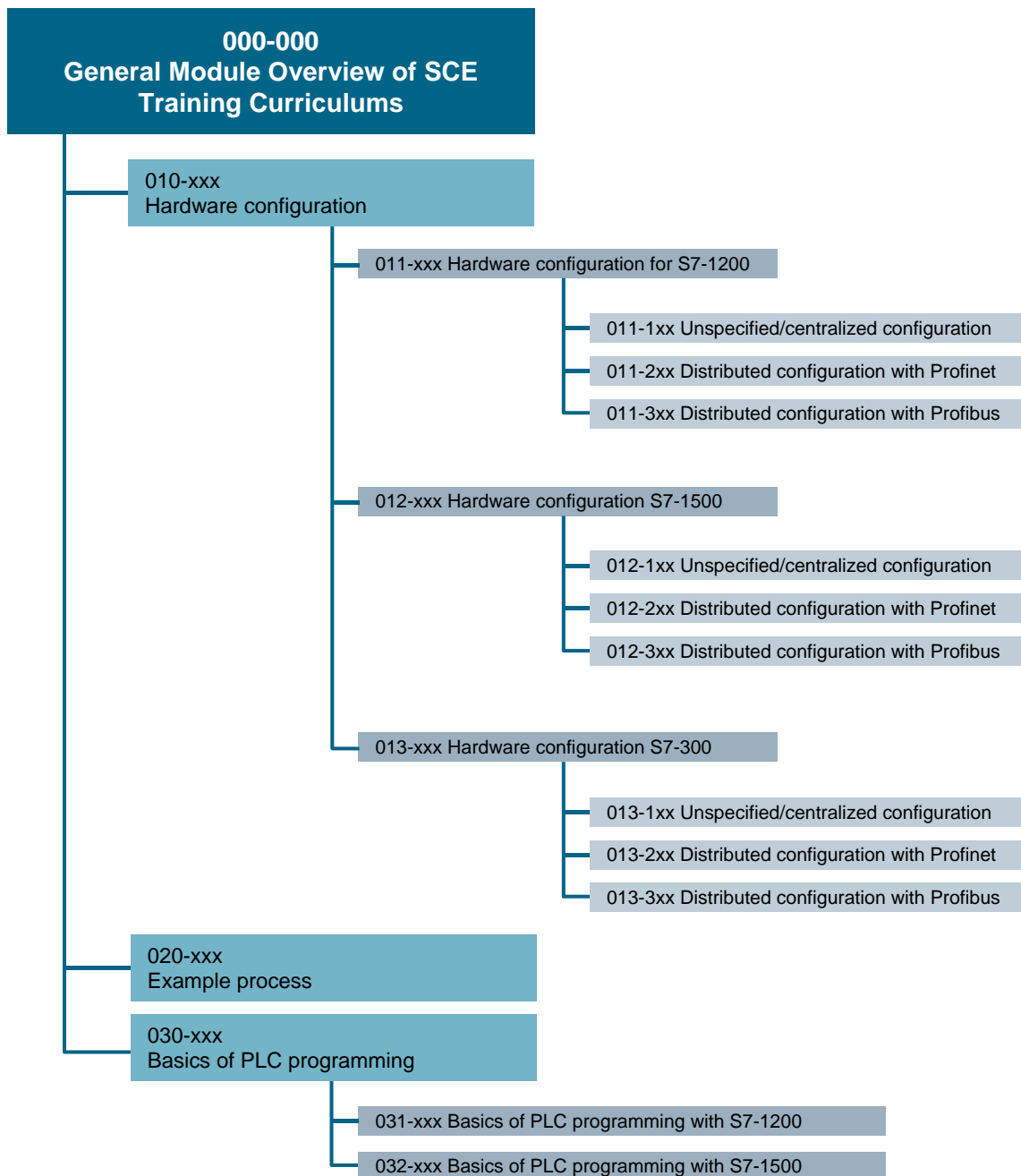


Figure 2: Basic modules

Because the hardware configuration can be performed and learned independent of a specific task, this topic has been placed first. The topic of hardware configuration is subdivided again into the hardware configuration of the various controllers that are available as SCE Trainer Packages. These currently include the S7-1200, S7-1500 and S7-300. There is a further subdivision into different architectures or types of configuration below each of the different controllers. Specifically, these currently consist of the unspecified/centralized configuration, distributed configuration with PROFIBUS and distributed configuration with PROFINET. Other architectures are possible as soon as new technologies are foreseeable. The above-described outline is shown in Figure 2. In Figure 3, the chapters with complete module number highlighted in green represent chapters already prepared.

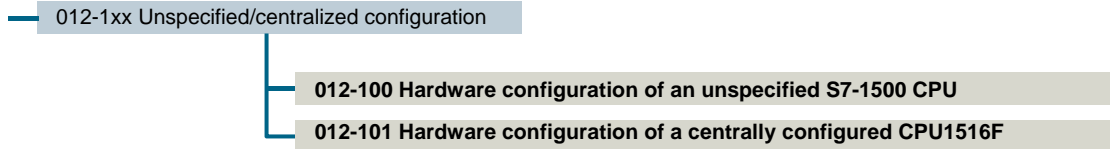


Figure 3: Completed chapters of the "Hardware configuration – S7-1500" topic area

The "Hardware configuration" topic is followed by the "Example process" topic. This topic is not a training unit. Rather, it describes example processes that are to be used in the subsequent chapters for specific tasks. To date, there is the sorting station example process. This will be used in the subsequent chapters on programming. Other example processes as well as additions to individual basic process descriptions are also conceivable here. The objective of SCE is to implement this example process with a SIMIT model so that learners can test their implementation with a simulated process.



Figure 4: "Example processes" topic area with completed process description

The "Basics of PLC programming" topic now follows, which is again subdivided into the S7-1200 and S7-1500 controllers to make it easier to get started. The S7-300 controller is not mentioned separately here since its programming is basically the same as that of the S7-1500 with a few small differences. The subdivision below the controllers is identical in both cases. It starts with FC and FB programming. Extremely simple tasks are provided here in each case to make it easy to get started. "Counters und timers", "Diagnostics", "Analog values" and "Global data blocks" round out this topic area. The addition of other topics is also conceivable here. Some topics are also categorized under the "Advanced programming" topic area. Refer to the next section for more information regarding this. Figure 5 summarizes the above-described outline once more.

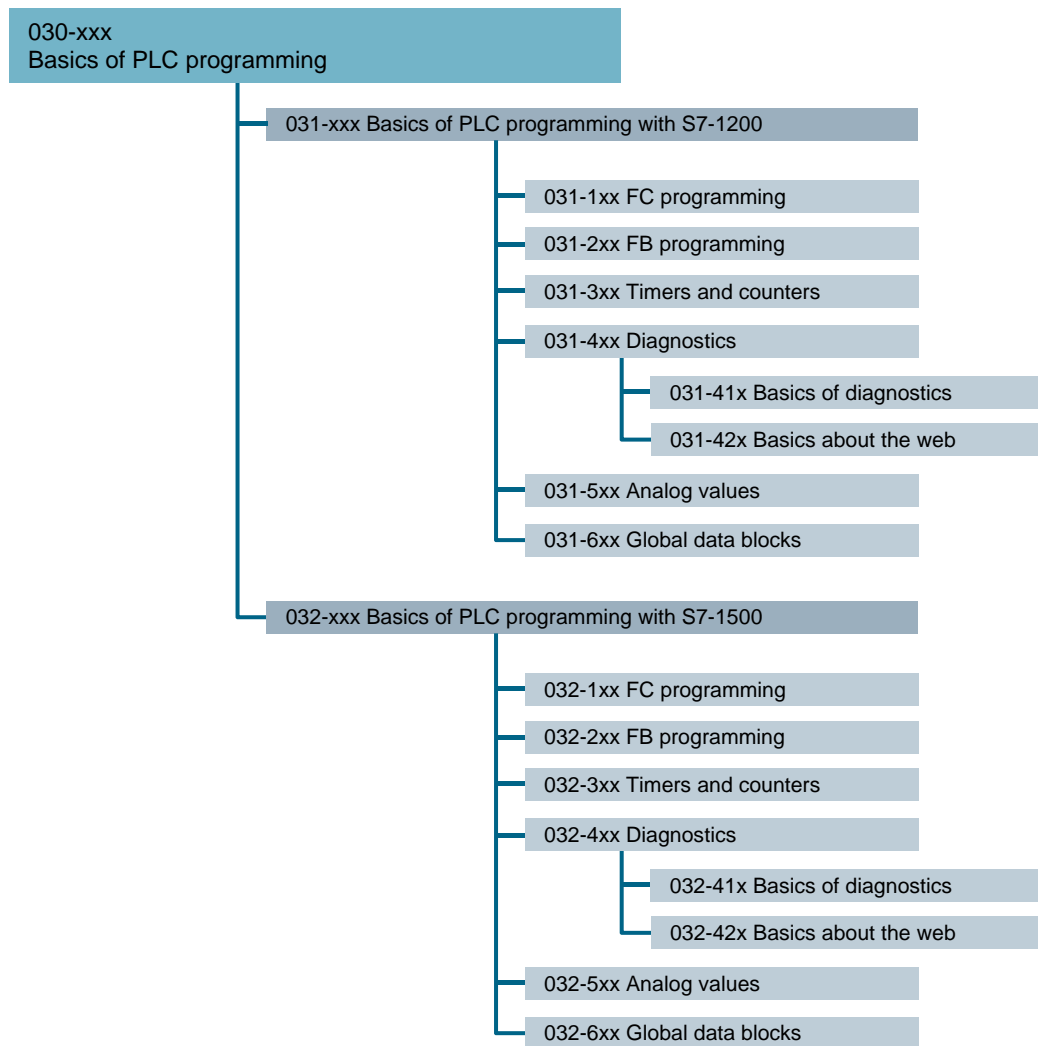


Figure 5: Structure of "Basics of PLC programming"

1.3 Advanced modules

Advanced modules include "Visualization", "Advanced programming" and "Drives", as shown in Figure 6. There is also a subdivision into the S7-1200 and S7-1500 controllers here, but the contents with S7-1500 will be preferably shown here. Contents for S7-1200 are available only when they differ significantly from that of the S7-1500. Therefore, these subdivisions are also marked with an asterisk (*).

The "Visualization" topic is only indirectly dependent on the controller. The utilized panel or PC is actually the deciding factor here. Within the visualization hardware segments (panel, PC), different chapters are offered, for example, for adding a panel and its configuration, simple visualizations and animations.

The "Advanced programming" topic includes programming topics for advanced learners, which are taught mainly using the S7-1500. Learners should be able to transfer the contents to other controllers on their own at this point. For this reason, topics are only included for S7-1200 when there are significant differences.

The "Drives" topic will also be handled similarly. Here, there is a further subdivision as to whether drives are connected via PROFIBUS or PROFINET.

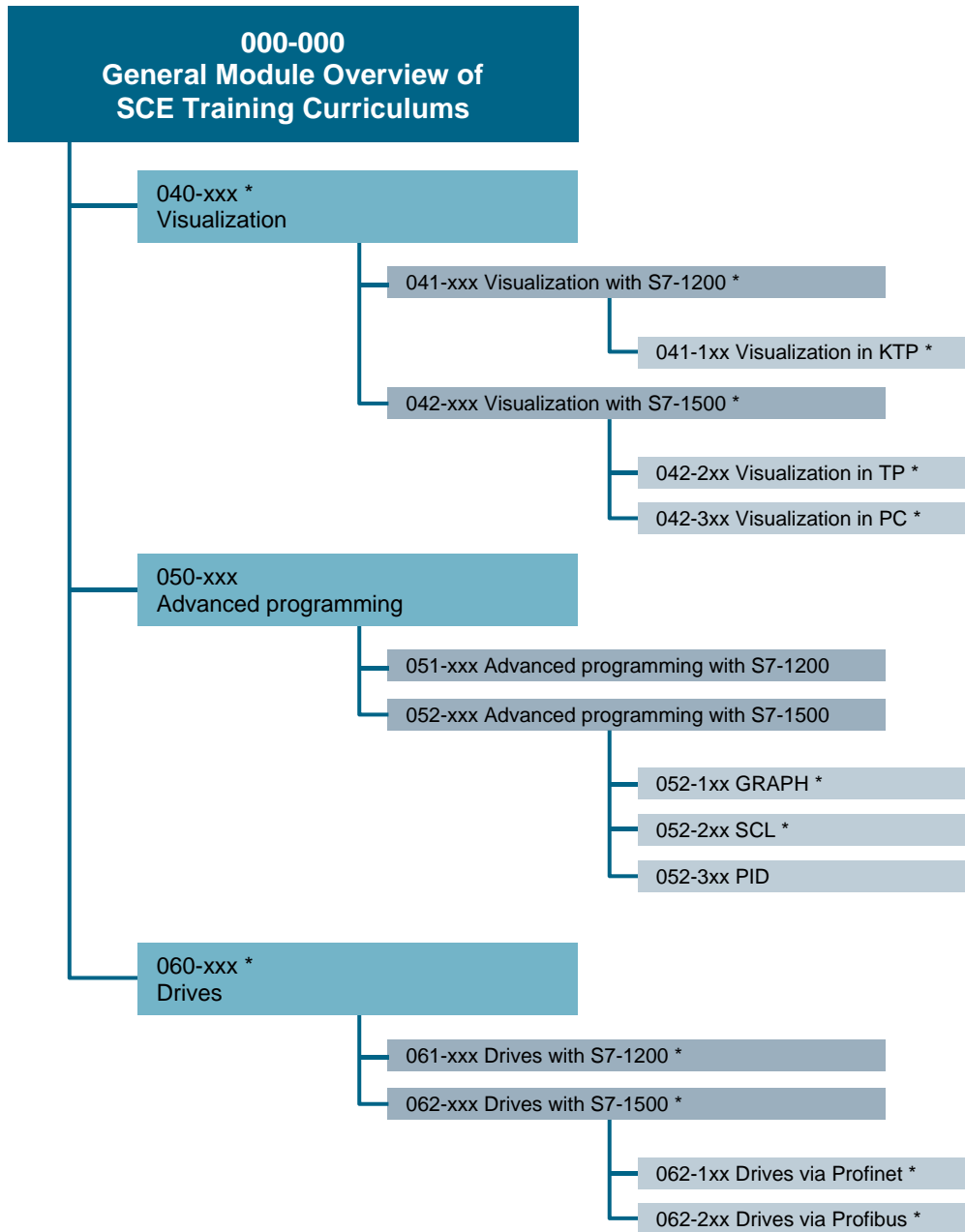


Figure 6: Advanced modules

* Upcoming

1.4 Special modules

Special modules include all modules shown in Figure 7. These modules cover general topics such as safety, security and energy efficiency as well as specific technologies such as RFID and Siwarex.

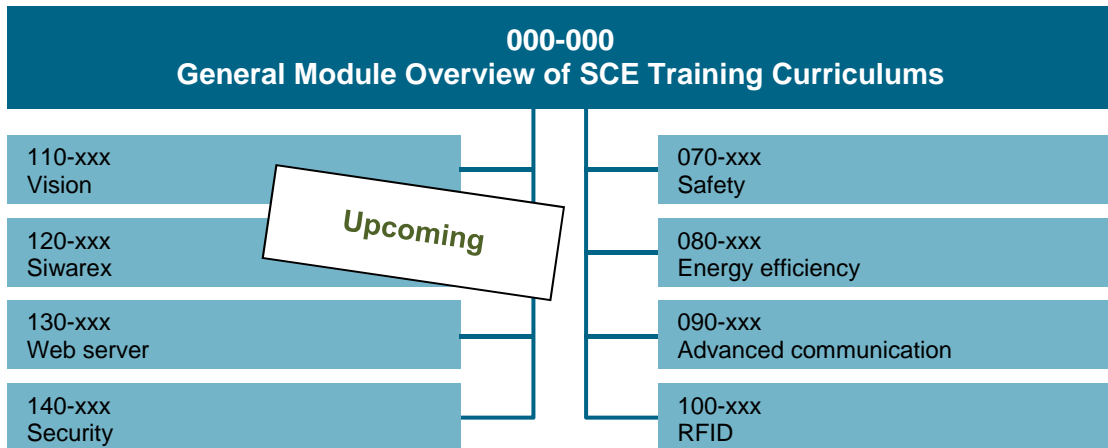


Figure 7: Special modules

"Advanced communication" includes all communication topics involving communication between controllers and between controllers and other systems via PROFIBUS, PROFINET, OPC UA and alternative bus systems. Of course, topics for plant-wide communication and wireless communication can also be found here. Figure 8 shows the current structure.

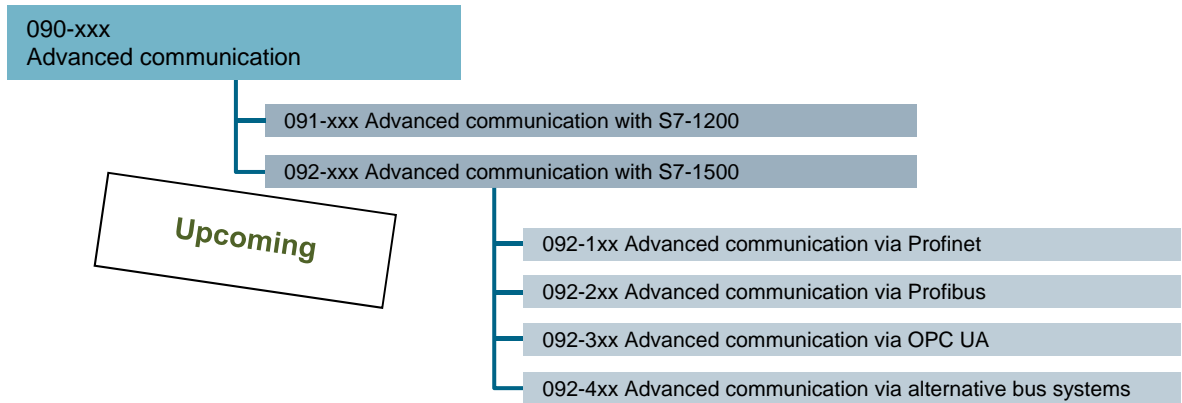


Figure 8: "Advanced communication" topic

2 Concept description

2.1 Material at a glance

The above-described modules will be made available as SCE Training Curriculums. Additional SCE training material is available to assist in lesson planning or self-study. This includes the SIMIT model, which implements the example process as a simulation. The simulation can already be started with a demo version of SIMIT, which can be used to check your own programming. As a result, a real plant is not needed. If you would rather work with real plants, you can build one yourself based on the description of the example process. SCE currently does not offer a real model of the example process. Furthermore, presentations are offered that contain a brief introduction to the contents of each chapter and are thus ideally suited for inclusion in the lesson. Of course, these presentations can also be used in self-study. Model solutions/projects are another important component of the SCE learning materials. They allow you to compare your own solution but are also suited to working on just a few topics and building on a particular model solution. Advanced materials such as videos and animations are integrated as links within the SCE training curriculum. These are also available via the SCE website or YouTube.

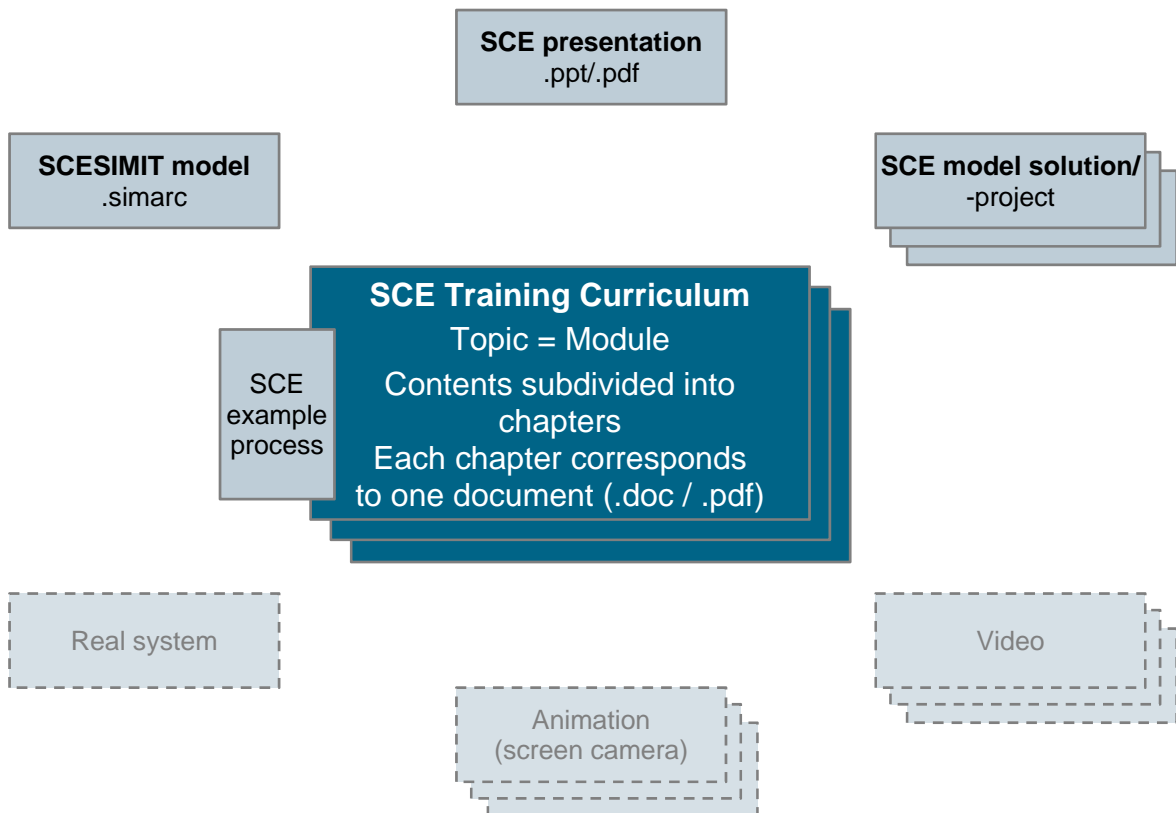


Figure 9: SCE training material

2.2 Structure of the individual chapters based on the teaching methodology

The core of the training materials is the SCE training curriculums in which each chapter forms a self-contained training unit. As shown in Figure 10, the chapters always start with a goal. The theory deemed as relevant is presented in the next section. A specific task is then formulated, which is next planned and implemented using an example. This is rounded off with a checklist that provides support when testing the implementation according to structured step-by-step instructions. The exercise follows, beginning with another task that is now to be planned and implemented independently. A checklist at the end enables learners to check their own solution.

The individual chapters can be worked on as modular units. Certain previous chapters must be built upon for this. Information on which chapters serve as the basis for a particular chapter can be found at the start of every document under "Prerequisites".

The chapters are structured within the above-described concept.

The example process is to accompany the learners continuously so that understanding of the actual process does not command too much attention and the focus can instead be on the actual training contents. At the same time, the newly added "Planning" section should be helpful by acting as an intermediary between the task and solution and a guide for planning the implementation. The step-by-step instructions have also been changed in the current concept. They are now structured. This helps beginning learners to obtain an overview of what they are doing and enables advanced learners to skip certain individual steps when they are already familiar with the action. In this way, the instructions can be worked through on a highly individual basis depending on the level of knowledge. The final checklist now also contains information about what can be tested and what needs to work with the implemented solution. Ideally, if an item of the checklist is not completed, information on the section from which this error might have originated can also be provided.

1

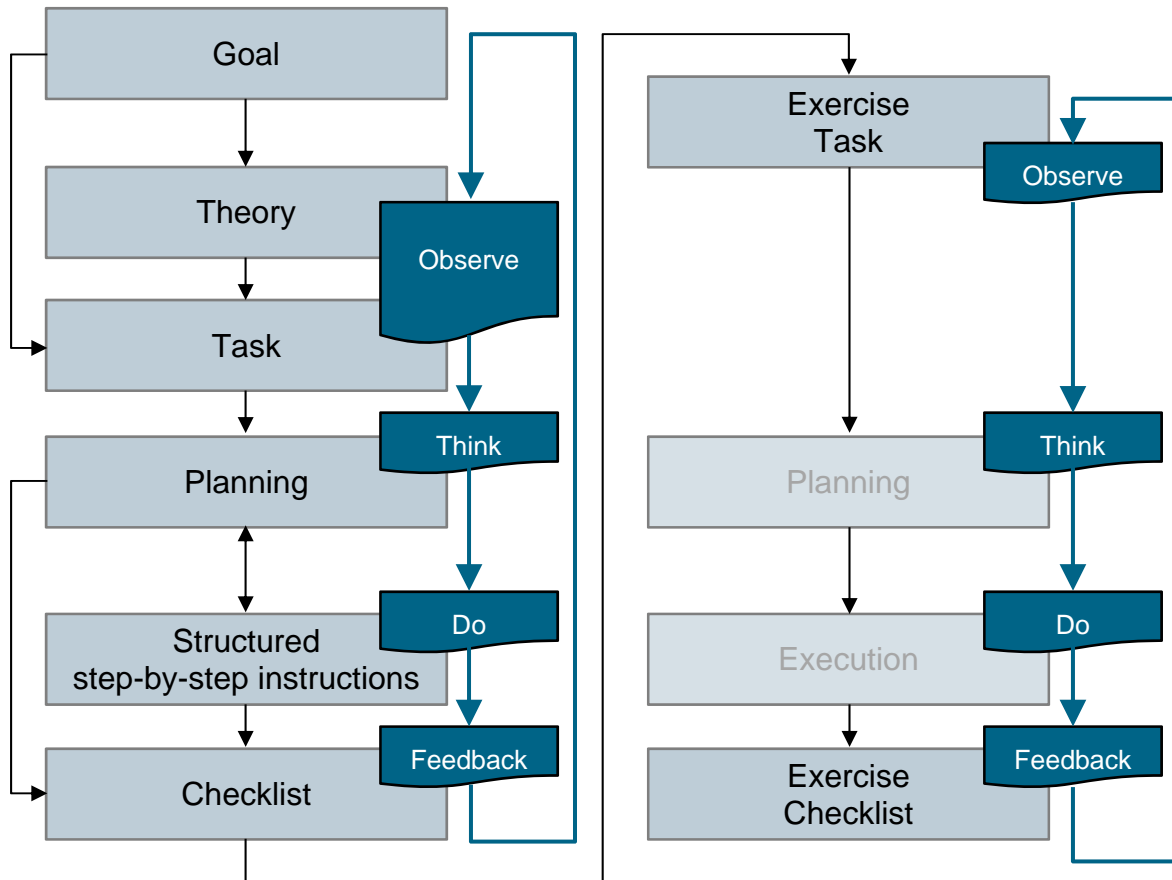


Figure 10: Structure of teaching methodology

The purpose of the exercise is to allow independent execution of a similar task. Only the task is specified and planning and implementation (= execution) must take place independently. This can also be used as a supplemental task for adept learners who are faster than the rest of the class or for self-study. A checklist at the end of the exercise is used to check the implemented solution. The model project can also be used for comparison of the solution. The model solution (= model project) is provided for each chapter and contains the result of the structured step-by-step instructions and the exercise. Depending on what the chapter is based on, the model solution obviously also contains other solutions.

3 Additional information

You can find additional information on the TIA Portal modules as an orientation aid for initial and advanced training.

www.siemens.com/sce/tia

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7 CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, get in touch with your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	5
2	Prerequisite.....	5
3	Theory.....	5
3.1	SIMATIC S7-1500 automation system	5
3.1.1	Range of modules	7
3.1.2	Example configuration	10
3.2	Operator control and display elements of the CPU 1516F-3 PN/DP	11
3.2.1	Front view of the CPU 1516F-3 PN/DP with integrated display	11
3.2.2	Status and error displays.....	11
3.2.3	Operator control/connection elements of the CPU 1516F-3 PN/DP behind the front flap ..	12
3.2.4	SIMATIC memory card	13
3.2.5	Mode switch.....	13
3.2.6	Display of the CPU	14
3.3	Memory areas of the CPU 1516F-3 PN/DP and the SIMATIC memory card	16
3.4	STEP 7 Professional V13 (TIA Portal V13) programming software	17
3.4.1	Project	18
3.4.2	Hardware configuration	18
3.4.3	Central and distributed automation structure	19
3.4.4	Planning the hardware.....	19
3.4.5	TIA Portal – Project view and portal view.....	20
3.4.6	Basic settings for the TIA Portal	22
3.4.7	Setting the IP address on the programming device	24
3.4.8	Setting the IP address in the CPU.....	27
3.4.9	Formatting the memory card in the CPU	30
3.4.10	Resetting the CPU to factory settings.....	31
4	Task	32
5	Planning.....	33

6	Structured step-by-step instructions	34
6.1	Create a new project	34
6.2	Read the hardware of the SIMATIC S7-1500	35
6.3	Configure the Ethernet interface of the CPU 1516F-3 PN/DP	41
6.4	Configure the access level for the CPU 1516F-3 PN/DP	42
6.5	Insert power module PM 190W 120/230VAC	43
6.6	Configure the address areas of the digital input and output modules	44
6.7	Save and compile the hardware configuration	45
6.8	Download the hardware configuration to the device	46
6.9	Archive the project	51
6.10	Checklist	52
7	Additional information	53

UNSPECIFIED HARDWARE CONFIGURATION – FOR A SIMATIC S7-1500

1 Goal

In this chapter, you will first learn how to **create a project**. Next you will be shown in one part of the task how you can use the **TIA Portal** to detect **hardware** already installed and add it to a project. This hardware will then be configured.

2 Prerequisite

You do not need any previous knowledge from other chapters to successfully complete this chapter. You only need an S7-1500 controller.

3 Theory

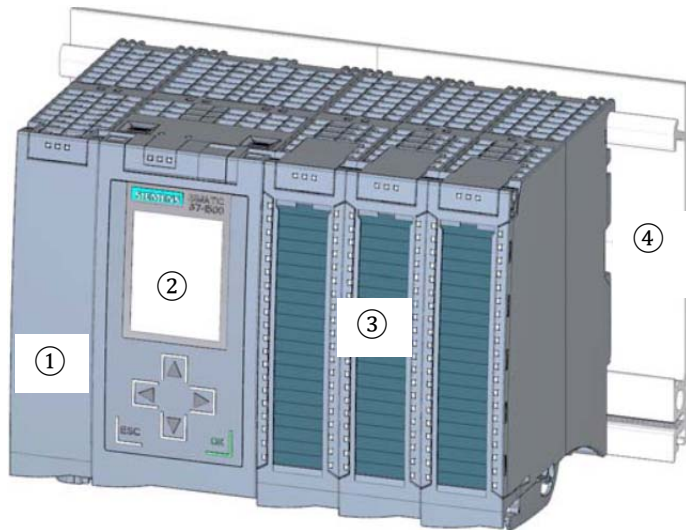
3.1 SIMATIC S7-1500 automation system

The SIMATIC S7-1500 automation system is a modular controller system for the middle to upper performance range. A comprehensive range of modules is available to optimally adapt the system to the automation task.

SIMATIC S7-1500 is the next generation of the SIMATIC S7-300 and S7-400 automation systems with the following new performance features.

- Increased system performance
- Integrated motion control functionality
- PROFINET IO IRT
- Integrated display for machine-level operation and diagnostics
- STEP 7 language innovations while maintaining proven functions

2 The S7-1500 controller consists of a power supply ①, a CPU with integrated display ② and input and output modules for digital and analog signals ③. The modules are mounted on a mounting rail with integrated DIN rail profile ④. If necessary, communication processors and function modules are also used for special tasks such as stepper motor control.



The programmable logic controller (PLC) uses the S7 program to monitor and control a machine or process. In doing so, the S7 program scans the I/O modules via input addresses (%I) and addresses their output addresses (%Q).

The system is programmed with the STEP 7 Professional V13 software.

3.1.1 Range of modules

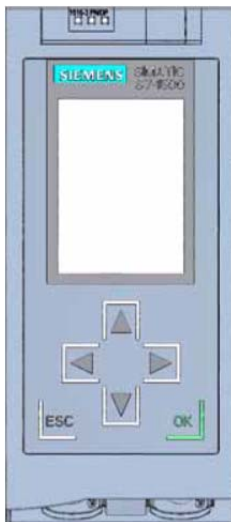
The SIMATIC S7-1500 is a modular automation system and offers the following range of modules:

Central processing units (CPUs) with integrated display

The CPUs have different performance capability and execute the user program. In addition, the other modules are supplied power via the backplane bus with the integrated system power supply.

Additional properties and functions of the CPU:

- Communication via Ethernet
- Communication via PROFIBUS/PROFINET
- HMI communication for HMI devices
- Web server
- Integrated technology functions (e.g. PID controller, motion control, etc.)
- System diagnostics
- Integrated security (e.g. know-how, copy, access, integrity protection)

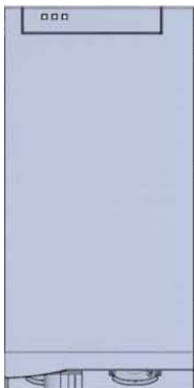


System power supply modules (PS) (rated input voltages 24 V DC to 230 V AC/DC)

with connection to the backplane bus supply the configured modules with the internal supply voltage.

**Load current supply modules (PM) (rated input voltages 120/230 V AC)**

do not have a connection to the backplane bus of the S7-1500 automation system. The load current supply is used to supply 24 V DC to the system power supply of the CPU, the input and output circuits of I/O modules and the sensors and actuators.



I/O modules

for digital input (DI) / digital output (DQ) / analog input (AI) / analog output (AQ)

**Technology modules (TM)**

as incremental encoders and pulse encoders with/without direction signal

**Communication modules (CM)**

for serial communication RS232 / RS422 / RS485, PROFIBUS and PROFINET



SIMATIC memory card

up to a maximum of 2 GB for storing program data and for easy replacement of CPUs during maintenance.



3.1.2 Example configuration

The following configuration of an S7-1500 automation system will be used for the program example in this curriculum.



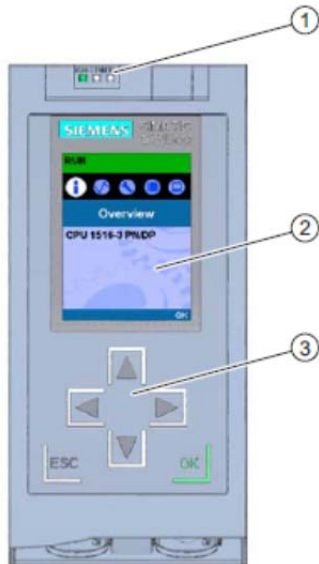
- ① Load current supply module (PM) with 120/230 V AC, 50 Hz / 60 Hz, 190 W input and 24 V DC / 8 A output
- ② Central processing unit CPU 1516F-3 PN/DP with integrated PROFIBUS and PROFINET interfaces
- ③ I/O module 32x digital input DI 32x24VDC HF
- ④ I/O module 32x digital output DQ 32x24VDC/0.5A HF
- ⑤ I/O module 8x analog input AI 8xU/I/RTD/TC ST
- ⑥ I/O module 4x analog output AQ 4xU/I ST

3.2 Operator control and display elements of the CPU 1516F-3 PN/DP

The figure below shows the operator control and display elements of a CPU 1516F-3 PN/DP

The arrangement and number of elements differ from this figure for other CPUs.

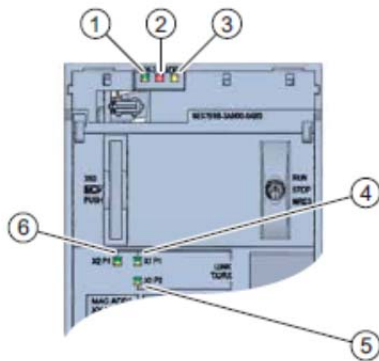
3.2.1 Front view of the CPU 1516F-3 PN/DP with integrated display



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Display
- ③ Control keys

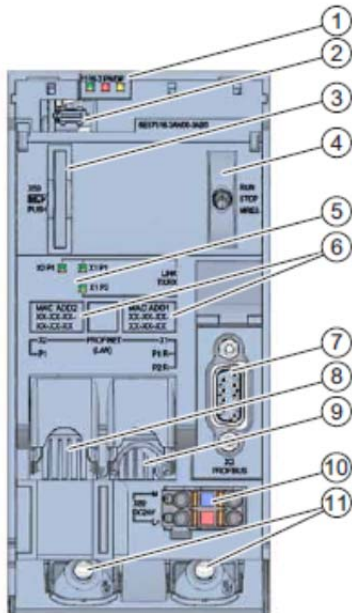
3.2.2 Status and error displays

The CPU comes with the following LED displays:



- ① RUN/STOP LED (yellow/green LED)
- ② ERROR LED (red LED)
- ③ MAINT LED (yellow LED)
- ④ LINK RX/TX LED for port X1 P1 (yellow/green LED)
- ⑤ LINK RX/TX LED for port X1 P2 (yellow/green LED)
- ⑥ LINK RX/TX LED for port X2 P1 (yellow/green LED)

3.2.3 Operator control/connection elements of the CPU 1516F-3 PN/DP behind the front flap



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Display connection
- ③ Slot for the SIMATIC memory card
- ④ Mode switch
- ⑤ LED displays for the 3 ports of the PROFINET interfaces X1 and X2
- ⑥ MAC addresses of the interfaces
- ⑦ PROFIBUS interface (X3)
- ⑧ PROFINET interface (X2) with 1 port
- ⑨ PROFINET interface (X1) with 2-port switch
- ⑩ Connection for supply voltage
- ⑪ Fastening screws

Note: The front flap with the display can be removed and inserted during operation.

3.2.4 SIMATIC memory card

A SIMATIC Micro Memory Card is used as the memory module for the CPUs. This is a preformatted memory card that is compatible with the Windows file system. It is available with various storage capacities and can be used for the following purposes:

- Transportable data storage medium
- Program card
- Firmware update card

The MMC **must** be inserted to operate the CPU as the CPUs have no integrated load memory. A commercially available SD card reader is needed to write/read the SIMATIC memory card with the programming device or PG/PC. This allows files to be copied directly to the SIMATIC memory card using Windows Explorer, for example.

Note: It is recommended that the SIMATIC memory card only be removed or inserted when the CPU is in the POWER OFF state.

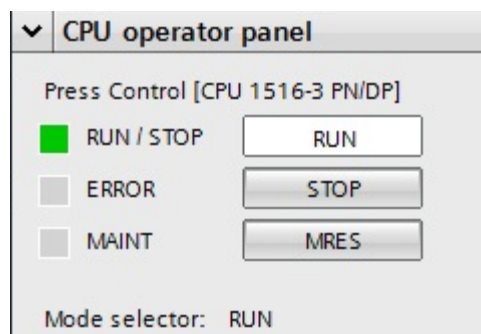
3.2.5 Mode switch

The mode switch allows you to set the operating mode of the CPU. The mode switch is designed as a toggle switch with 3 switch positions.

Position	Meaning	Explanation
RUN	RUN mode	The CPU processes the user program.
STOP	STOP mode	The CPU is not executing the user program.
MRES	Memory reset	Position for CPU memory reset.

You can also use the button on the CPU operator panel of the STEP 7 Professional V13 software in Online & Diagnostics to switch the operating mode (**STOP** or **RUN**).

The operator panel also contains an **MRES** button for performing a memory reset and displays the status LEDs of the CPU.



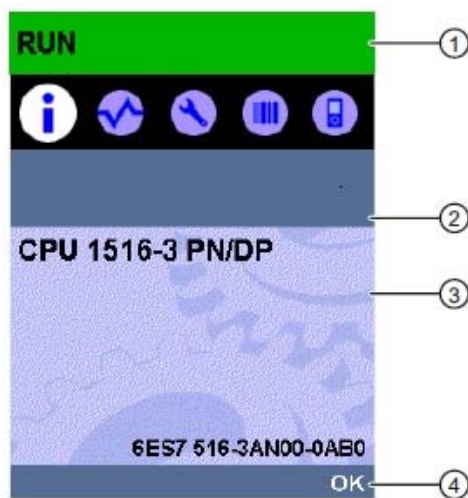
3.2.6 Display of the CPU

The S7-1500 CPU has a front flap with a display and control keys. Control data and status data can be displayed in various menus on the display and numerous settings can be configured. You use the control keys to navigate through the menus.

The display of the CPU offers the following functions:

- 6 different display languages can be selected.
- Diagnostic messages are displayed in plain text.
- The interface settings can be changed locally.
- Password assignment for display operation is possible through the TIA Portal.

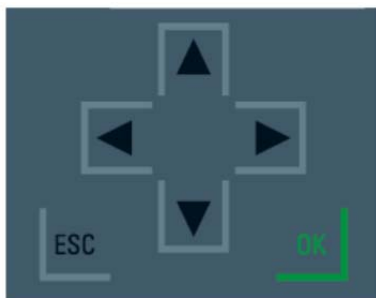
View of the display of an S7-1500:



- ① CPU status information
- ② Submenu name
- ③ Information display field
- ④ Navigation aid, e.g. OK/ESC or the page number

Control keys of the display






- Four arrow keys: "up", "down", "left", "right"
- An ESC key
- An OK key



Functions of the "OK" and "ESC" keys

- For menu commands in which an input can be made:
 - OK → valid access to the menu command, confirmation of input and exit from editing mode
 - ESC → restoration of original content (which means changes are not saved) and exit from editing mode
- For menu commands in which no input can be made:
 - OK → to next submenu command
 - ESC → back to previous menu command

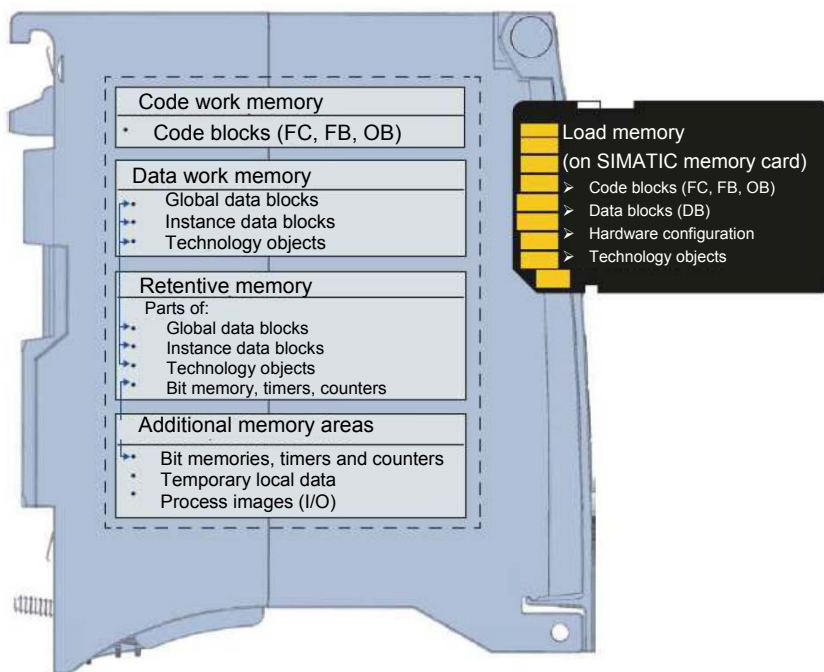
Available submenus of the display:

Main menu commands	Meaning	Explanation
	Overview	The "Overview" menu contains information about the properties of the CPU.
	Diagnostics	The "Diagnostics" menu contains information about diagnostic messages, the diagnostic description and the indication of interrupts. There is also information about the network properties of each interface of the CPU.
	Settings	In the "Settings" menu, the IP addresses of the CPU are assigned, the date, time, time zones, operating modes (RUN/STOP) and protection levels are set, the CPU memory is reset and its factory settings are restored and the status of firmware updates is displayed.
	Modules	The "Modules" menu contains information about the modules that are used in your configuration. The modules can be used as central or distributed modules. Distributed modules are connected to the CPU via PROFINET and/or PROFIBUS. You have the option here to set the IP addresses for a CPU.
	Display	In the "Display" menu, settings are made for all aspects of the display, such as the language setting, brightness setting and Energy-saving mode. (Energy-saving darkens the display. Standby mode switches off the display.)

3.3 Memory areas of the CPU 1516F-3 PN/DP and the SIMATIC memory card

The following figure shows the memory areas of the CPU and the load memory on the SIMATIC memory card.

In addition to the load memory, other data can be loaded onto the SIMATIC memory card using Windows Explorer. This includes recipes, data logs, project backups and additional documentation for the program.



Load memory

Load memory is non-volatile memory for code blocks, data blocks, technology objects and the hardware configuration. When these objects are downloaded to the CPU, they are first stored in the load memory. This memory is located on the SIMATIC memory card.

Work memory

Work memory is volatile memory that contains the code and data blocks. The work memory is integrated into the CPU and cannot be expanded. In S7-1500 CPUs, the work memory is divided into two areas:

- Code work memory:
 - The code work memory contains runtime-relevant parts of the program code.
- Data work memory:
 - The data work memory contains the runtime-relevant parts of the data blocks and technology objects.

At the operating mode transitions from POWER ON to startup and from STOP to startup, tags of global data blocks, instance data blocks and technology objects are initialized with their start values. Retentive tags retain their actual values that were saved in the retentive memory.

Retentive memory

Retentive memory is non-volatile memory for saving certain data in the event of power failure. The tags and operand areas that have been defined as retentive are saved in the retentive memory. This data is retained beyond power-off or power failure.

All other program tags are set to their start values at the operating mode transitions from POWER ON to startup and from STOP to startup.

The content of retentive memory is deleted by the following actions:

- Memory reset
- Reset to factory settings

Note: Certain tags of technology objects are also stored in the retentive memory. These tags are not deleted by a memory reset.

3.4 STEP 7 Professional V13 (TIA Portal V13) programming software

STEP 7 Professional V13 (TIA Portal V13) software is the programming tool for the following automation systems:

- SIMATIC S7-1500
- SIMATIC S7-1200
- SIMATIC S7-300
- SIMATIC S7-400
- SIMATIC WinAC

STEP 7 Professional V13 provides the following functions for plant automation:

- Configuration and parameter assignment of the hardware
- Specification of the communication
- Programming
- Testing, commissioning and servicing with operational/diagnostic functions
- Documentation
- Creation of visualizations for SIMATIC Basic Panels using the integrated WinCC Basic software
- Visualization solutions for PCs and other panels can also be created with other WinCC software packages

Support is provided for all functions through detailed online help.

3.4.1 Project

To implement a solution for an automation and visualization task, you create a project in the TIA Portal. A project in the TIA Portal contains the configuration data for the configuration and internetworking of devices as well as the programs and the configuration of the visualization.

3.4.2 Hardware configuration

The *hardware configuration* includes the configuration of the devices, consisting of the hardware of the automation system, the intelligent field devices and the hardware for visualization. The configuration of the networks specifies the communication between the various hardware components. The individual hardware components are *inserted in the hardware configuration* from catalogs.

The hardware of automation systems comprises controllers (CPUs), signal modules for input and output signals (SMs) and communication processors, and interface modules (CP, IM). Power supply and voltage supply modules (PS, PM) are also available to supply the modules.

The signal modules and intelligent field devices connect the input and output data of the process to be automated and visualized to the automation system.

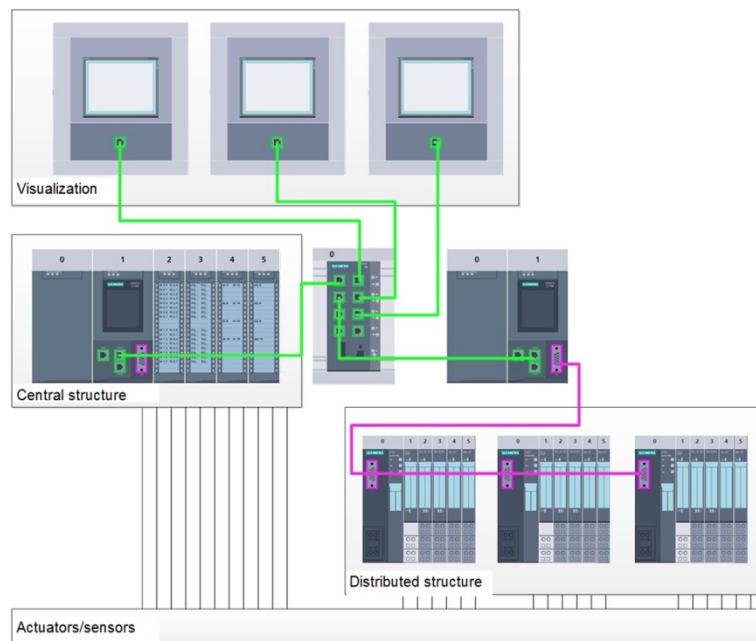


Figure 1: Example of hardware configuration with central and distributed structures

The hardware configuration enables the downloading of automation and visualization solutions to the automation system and access to the connected signal modules by the controller.

3.4.3 Central and distributed automation structure

Figure 1 shows an automation structure that contains both central and distributed structures.

In central structures, the input and output signals of the process are transmitted by way of conventional wiring to the signal modules, which are connected directly to the controller. Conventional wiring refers to the connection of sensors and actuators using 2-wire or 4-wire cables.

The distributed structure is the predominant structure used today. Here, the sensors and actuators are wired conventionally only as far as the signal modules of the field devices. The signal transmission from the field devices to the controller is implemented using an industrial communication system.

Both classic fieldbuses such as PROFIBUS, Modbus and Foundation Fieldbus as well as Ethernet-based communication systems such as PROFINET can be used as the industrial communication system.

In addition, intelligent field devices in which stand-alone programs run can also be connected via the communication system. These programs can also be created with the TIA Portal.

3.4.4 Planning the hardware

Before you can configure the hardware, you must plan it (hardware planning). In general, you begin by selecting which controllers are needed and how many. Next you select the communication modules and signal modules. The selection of signal modules is based on the number and type of inputs and outputs needed. As the final step, a power supply that ensures that the necessary power is supplied must be selected for each controller or field device.

The functionality required and the ambient conditions are of vital importance for planning the hardware configuration. For example, the temperature range in the application area sometimes limits the devices available for selection. Fail-safe operation might be another requirement, for example.

The [TIA Selection Tool](#) (Select automation technology → TIA Selection Tool and follow the instructions) provides you support. Note: TIA Selection Tool requires Java.

Note for online research: *If more than one manual is available, you should look for the description "Device Manual", "Product Manual" or simply "Manual" (as opposed to "Function Manual", "List Manual", "System Manual", etc.) in order to find the device specifications.*

3.4.5 TIA Portal – Project view and portal view

The TIA Portal has two important views. When started, the TIA Portal displays the portal view by default. This view makes getting started easier, especially for beginning users.

The portal view provides a task-oriented view of the tools for working on the project. Here, you can quickly decide what you want to do and open the tool for the task at hand. If necessary, a change to the project view takes place automatically for the selected task.

Figure 2 shows the portal view. At the bottom left, there is an option to switch between this view and the project view.

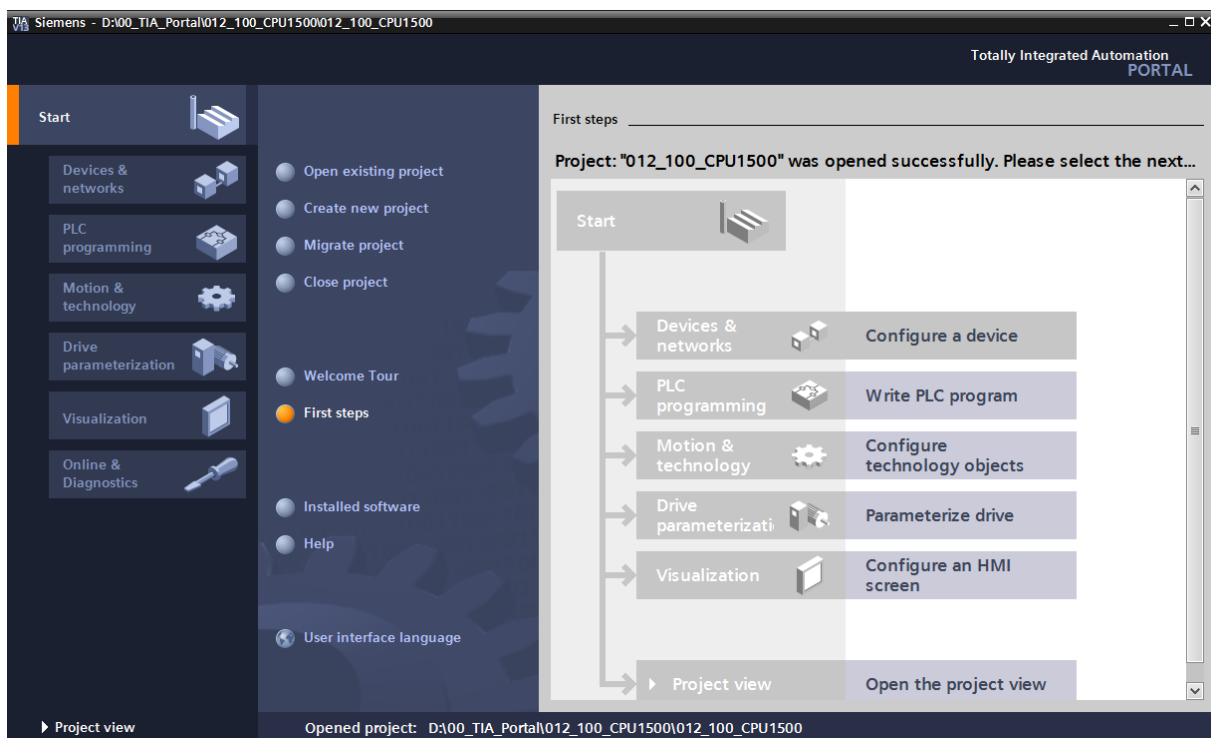


Figure 2: Portal view

The project view, as shown in Figure 3, is used for hardware configuration, programming, creation of the visualization and many other tasks.

By default, the project view displays the menu bar with the toolbars at the top, the project tree with all components of a project on the left and the so-called task cards with instructions and libraries, for example, on the right.

If an element (for example, the device configuration) is selected in the project tree, it is displayed in the center and can be worked on there.

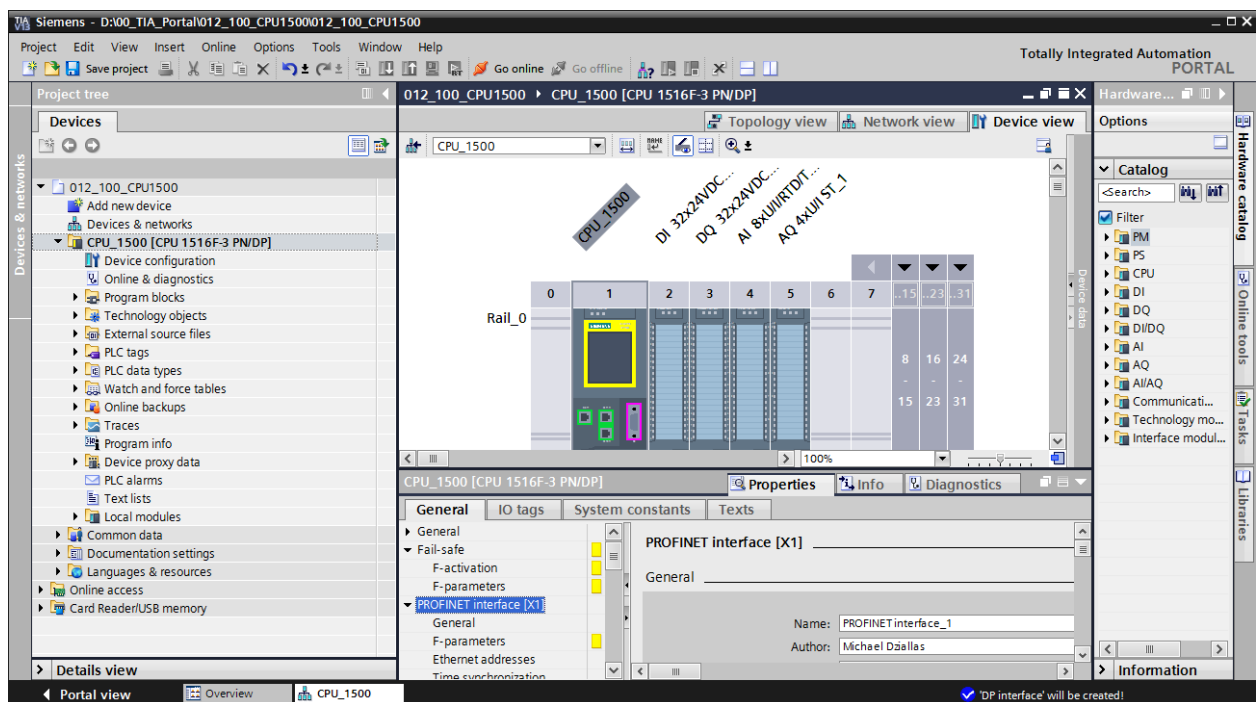
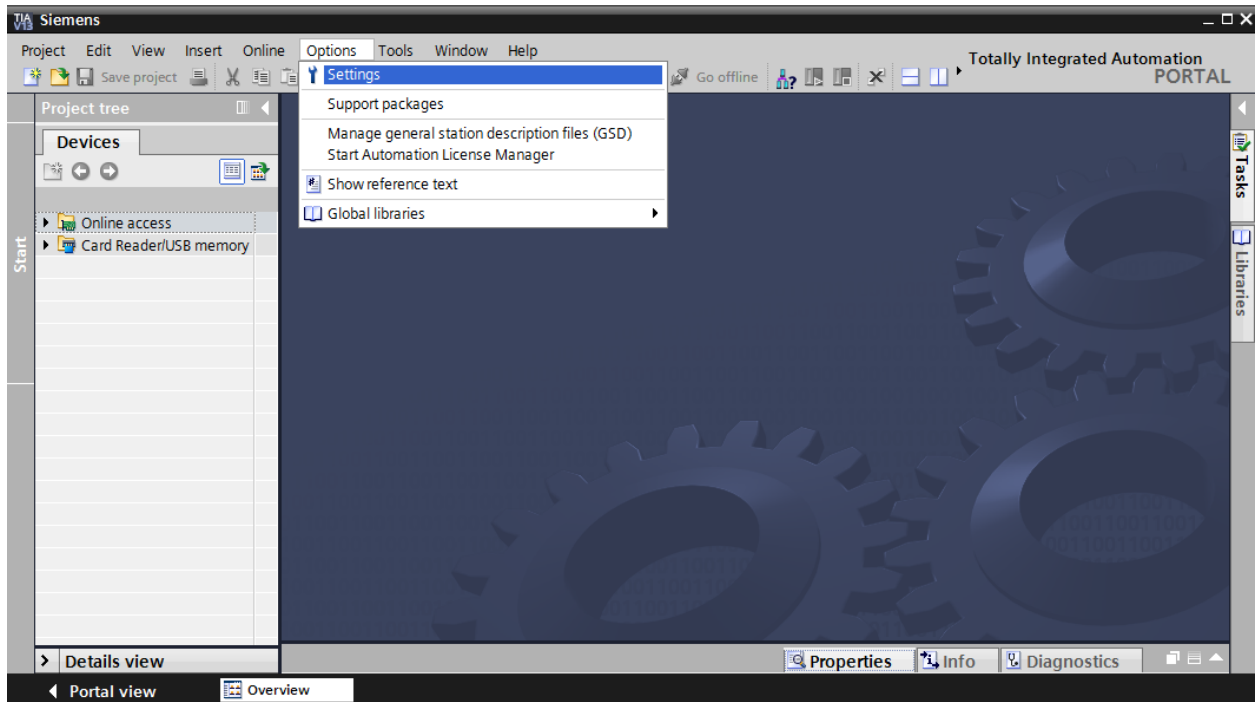


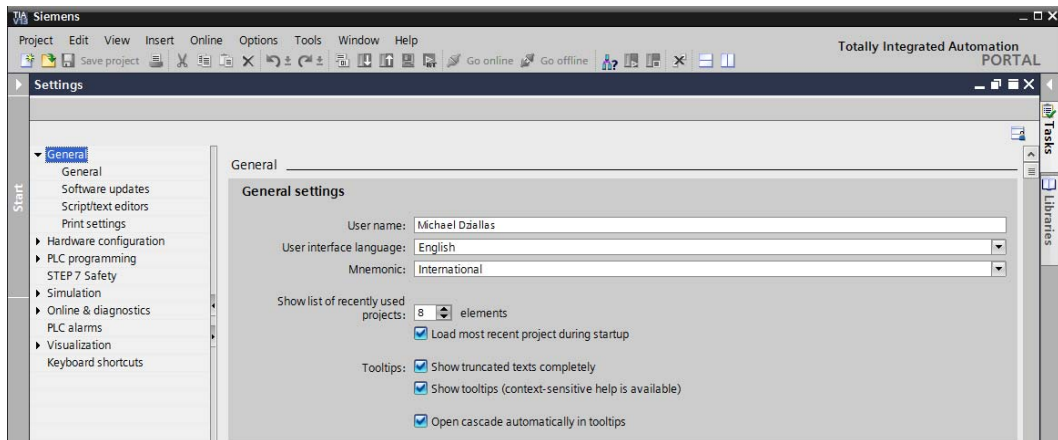
Figure 3: Project view

3.4.6 Basic settings for the TIA Portal

- Users can specify their own default settings for certain settings in the TIA Portal. A few important settings are shown here.
- In the project view, select the →"Options" menu and then → "Settings".

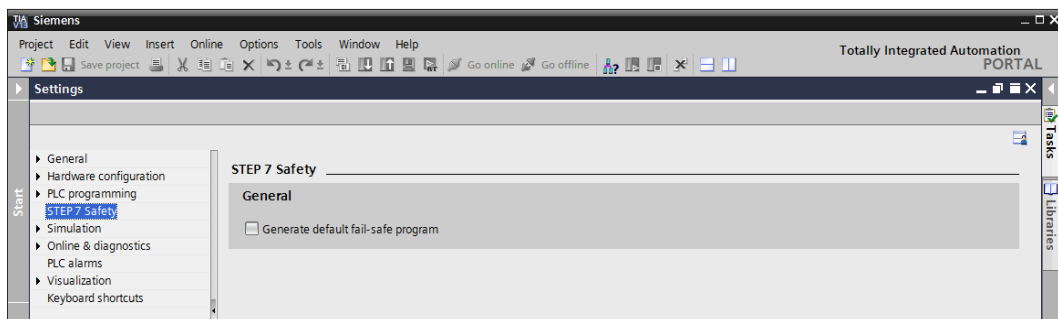


- One basic setting is the selection of the user interface language and the language for the program display. In the curriculums to follow, "English" will be used for both settings.
- Under → "General" in "Settings", select "User interface language → English" and "Mnemonic → International".



Note: These settings can always be changed.

- When Safety CPUs are used (e.g. CPU 1516F-3 PN/DP) without the use of safety engineering, it is recommended that automatic creation of the safety program be deactivated before creating a project.
- In "Settings" under the → "STEP 7 Safety" item, deactivate → "Generate default fail-safe program".




3.4.7 Setting the IP address on the programming device

2

To program SIMATIC S7-1500 from the PC, the programming device or a laptop, you need a TCP/IP connection or an optional PROFIBUS connection.

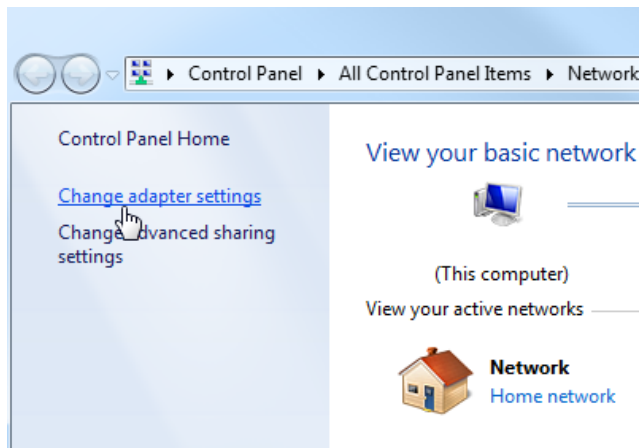
For the PC and SIMATIC S7-1500 to communicate with each other via TCP/IP, it is important that the IP addresses of both devices match.

First, we show you how to set the IP address of a computer with the Windows 7 operating system.

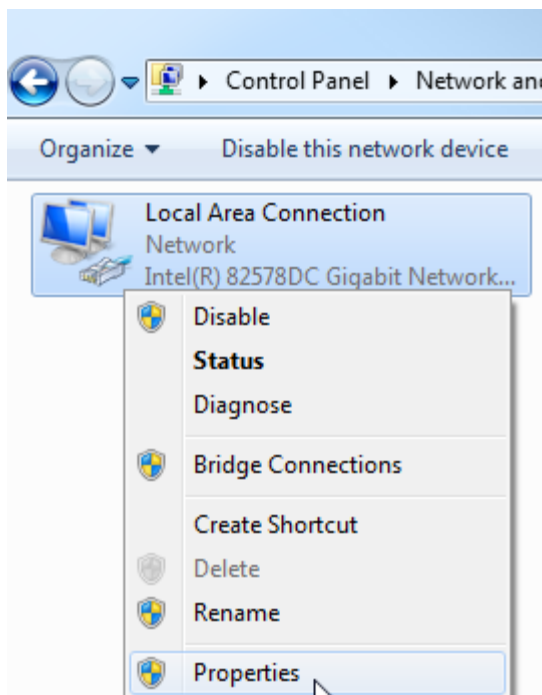
- Locate the network icon in the taskbar at the bottom  and click → "Open Network and Sharing Center".



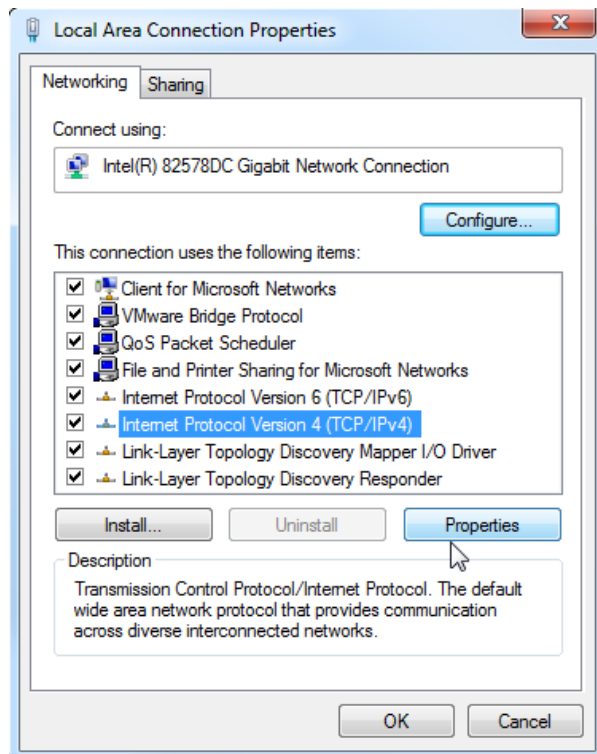
→ In the open Network and Sharing Center window, click → "Change adapter settings".



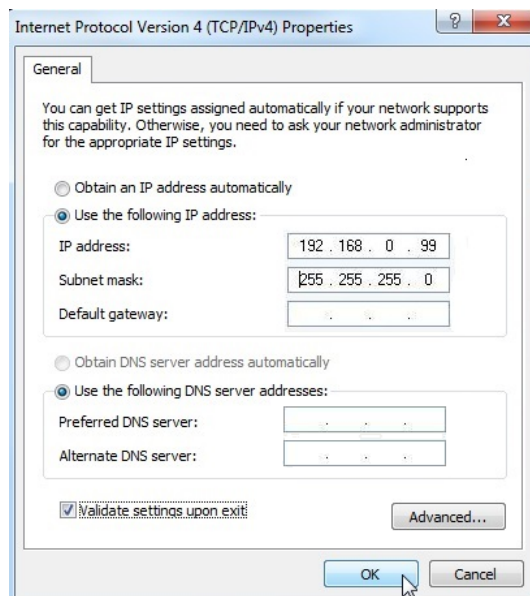
→ Select the desired → "Local Area Connection" that you want to use to connect to the controller and click → "Properties".



→ Next, select → "Properties" for → "Internet Protocol Version 4 (TCP/IP)".



→ You can use the following IP address, for example → IP address: 192.168.0.99 → Subnet mask 255.255.255.0 and accept the settings (→ "OK")



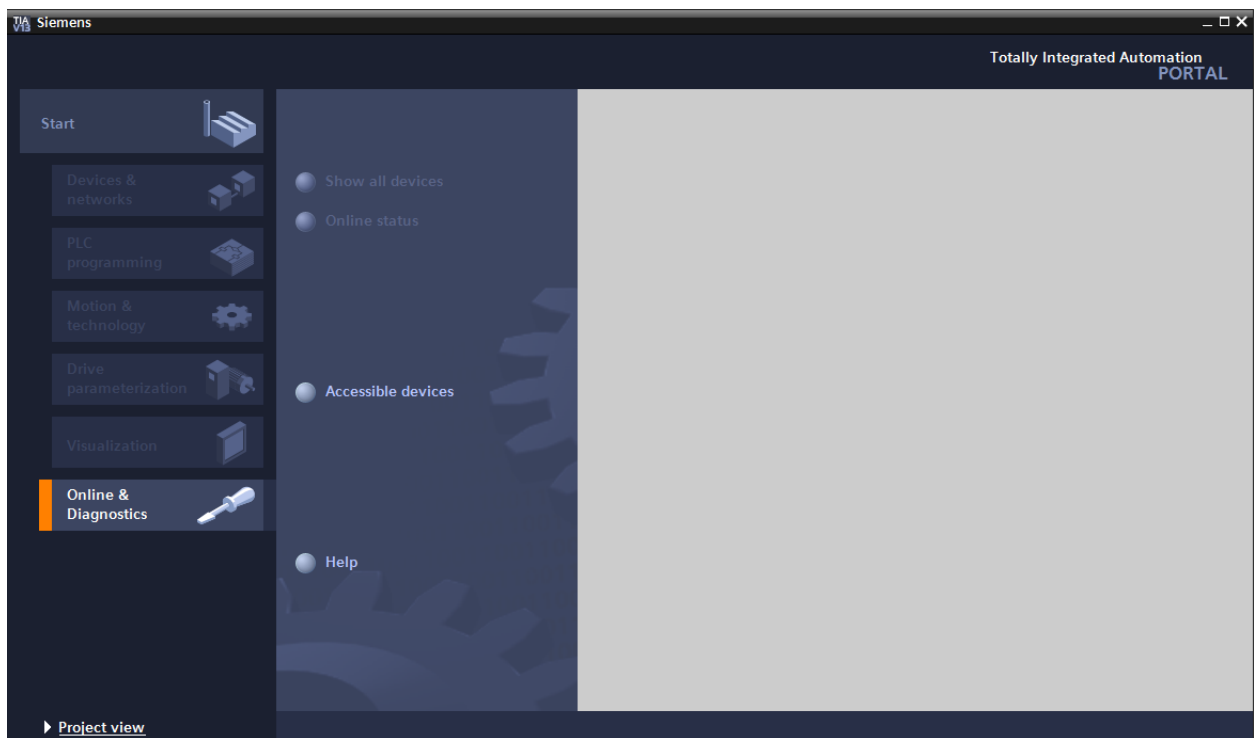
3.4.8 Setting the IP address in the CPU

The IP address of SIMATIC S7-1500 is set as follows.

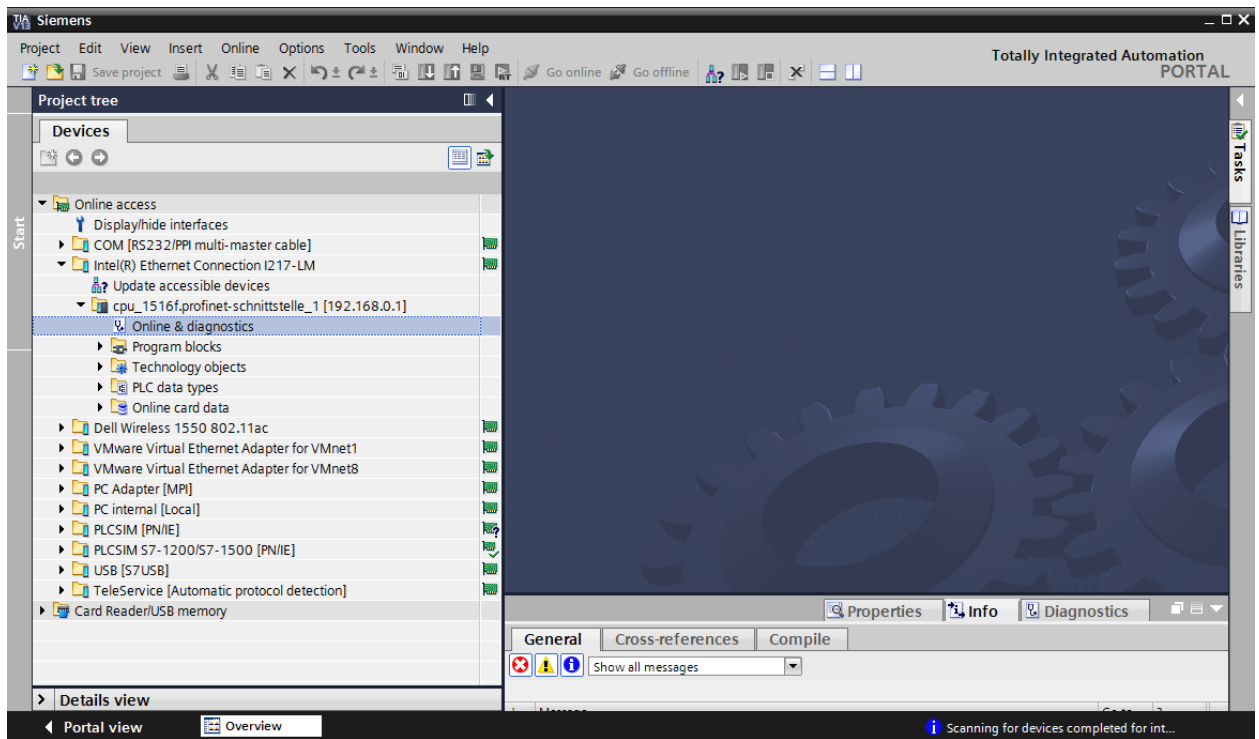
- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V13)



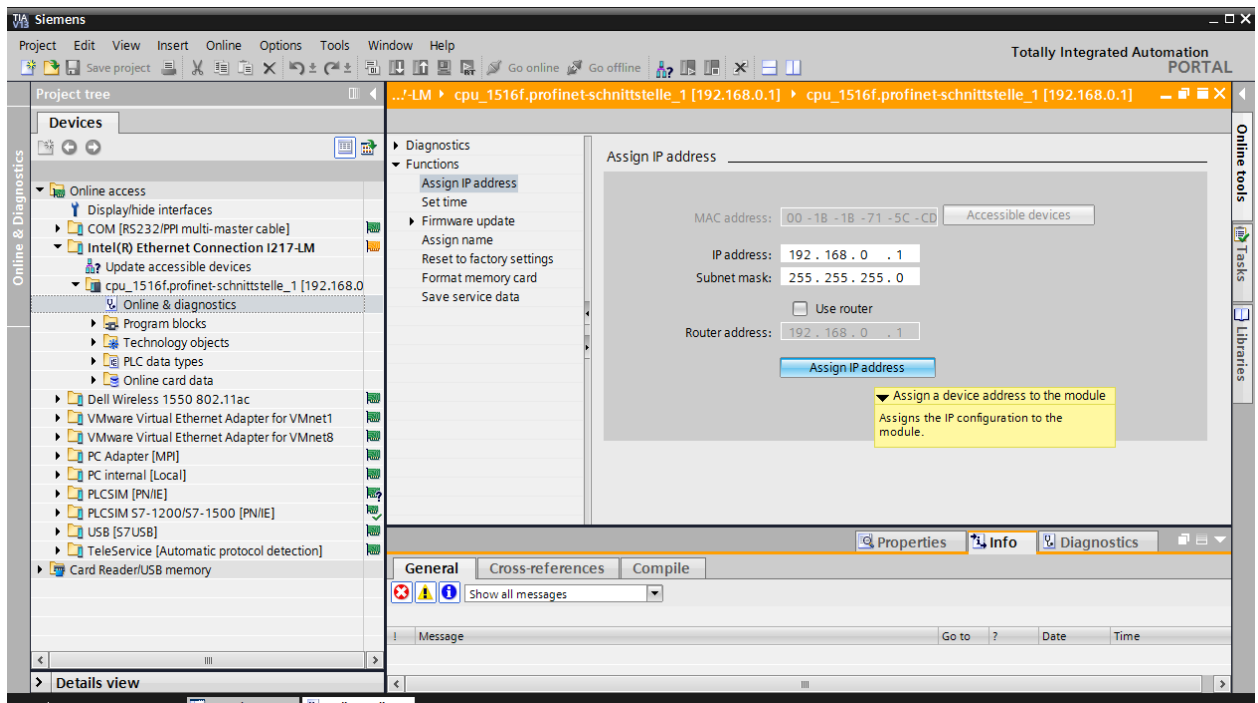
- Select → "Online & Diagnostics" and open the → "project view".



- In the project tree under → "Online access", select the network adapter that was set previously. If you click → "Update accessible devices" here, you will see the IP address (if previously set) or the MAC address (if IP address not yet assigned) of the connected SIMATIC S7-1500. Select → "Online & Diagnostics" here.

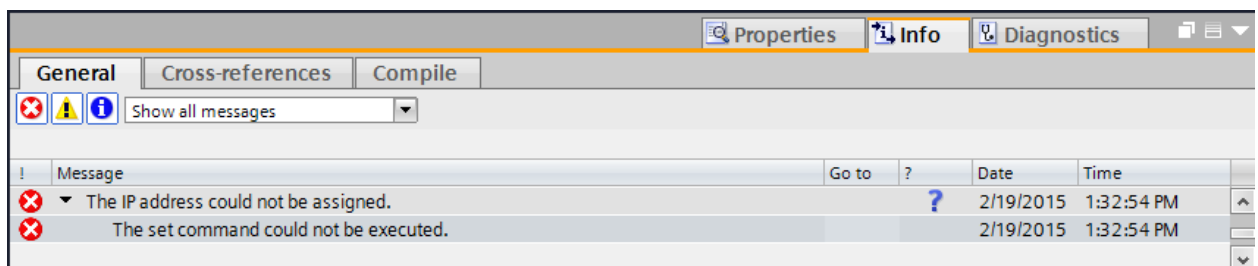


- Under → "Functions", you now find the → "Assign IP address" item. Enter the following IP address here (example): → IP address: 192.168.0.1 → Subnet mask 255.255.255.0. Next, click → "Assign IP address" and this new address will be assigned to your SIMATIC S7-1500.



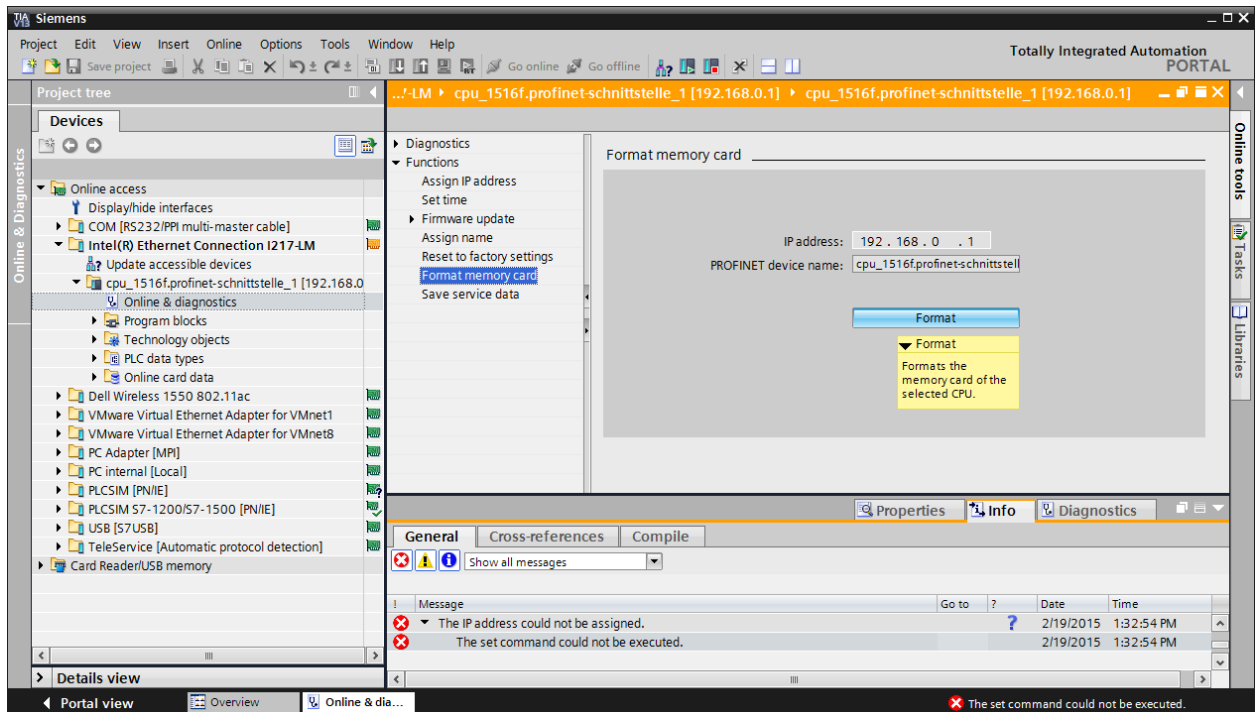
Note: The IP address of the SIMATIC S7-1500 can also be set via the display on the CPU, provided this is enabled in the hardware configuration.

- If the IP address was not successfully assigned, you will receive a message in the → "Info" window under → "General".

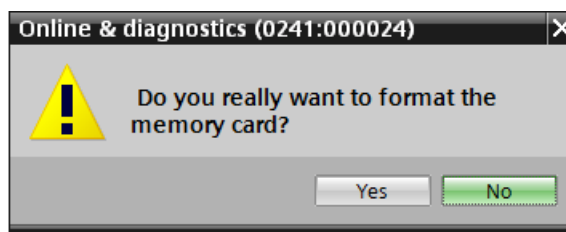


3.4.9 Formatting the memory card in the CPU

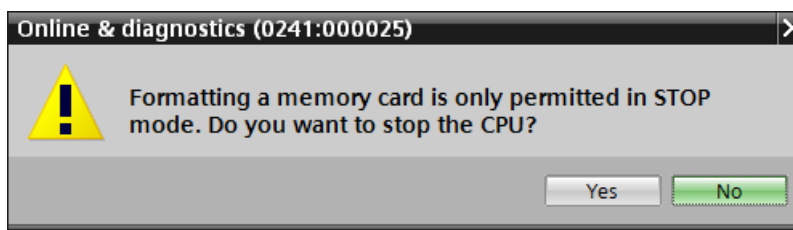
- If the IP address could not be assigned, the program data on the CPU must be deleted. This is accomplished in 2 steps: → "Format memory card" and → "Reset to factory settings".
- First, select the → "Format memory card" function and press the → "Format" button.



- Confirm the prompt asking if you really want to format the memory card with → "Yes".

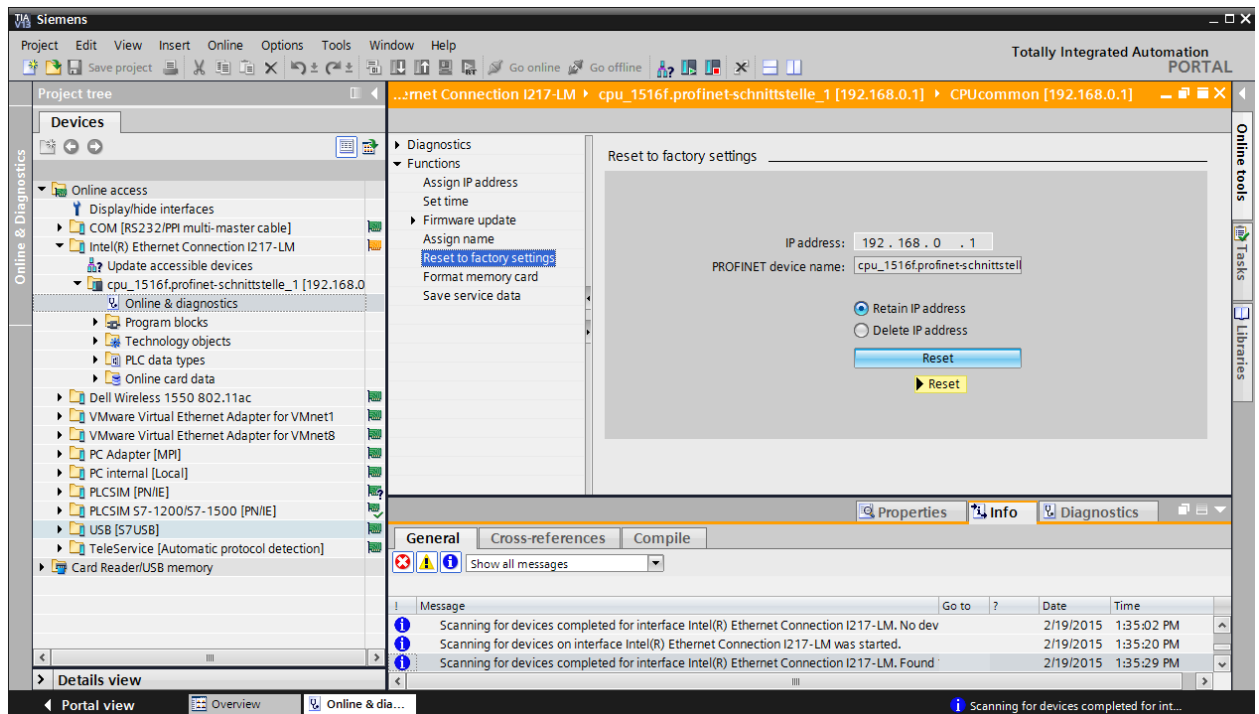


- If necessary, stop the CPU. (→ "Yes")

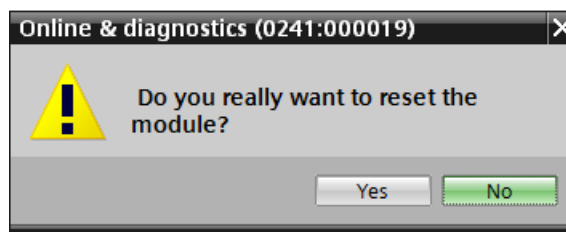


3.4.10 Resetting the CPU to factory settings

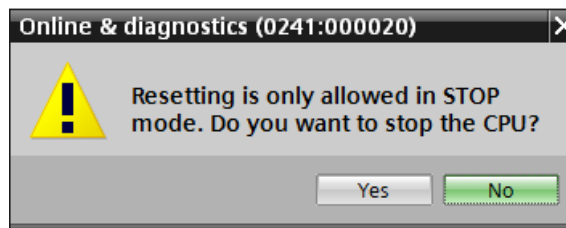
- Before you can reset the CPU, you must wait until the formatting in the CPU has finished. Then you must select → "Update accessible devices" and → "Online & diagnostics" of your CPU again. To reset the controller, select the → "Reset to factory settings" function and click → "Reset".



- Confirm the prompt asking if you really want to reset the module with → "Yes".



- If necessary, stop the CPU. (→ "Yes")



2

4 Task

Create a project and add the modules of the existing hardware (here: Trainer Package **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**) by using the automatic hardware detection of the **TIA Portal**. The following modules must be detected:

- SIMATIC S7-1500F, CPU 1516F-3 PN/DP, WORK MEMORY 1.5 MB PROGRAM, 5 MB DATA, 1. INTERFACE, PROFINET IRT WITH 2 PORT SWITCH, 2. INTERFACE, ETHERNET, 3. INTERFACE, PROFIBUS, 10 NS BITPERFORMANCE, SIMATIC MEMORY CARD REQUIRED (order number: 6ES7 516-3FN01-0AB0)
- 1X SIMATIC S7-1500, DIGITAL INPUT MODULE DI 32 X DC24V, 32 CHANNELS IN GROUPS OF 16 (order number: 6ES7521-1BL00-0AB0)
- 1X SIMATIC S7-1500, DIGITAL OUTPUT MODULE DQ 32 X DC24V / 0.5A; 32 CHANNELS (order number: 6ES7 522-1BL01-0AB0)
- 1X SIMATIC S7-1500, ANALOG INPUT MODULE AI 8 X U/I/RTD/TC, 16BIT RESOLUTION 8 CHANNELS IN GROUPS OF 8 (6ES7 531-7KF00-0AB0)
- 1X SIMATIC S7-1500, ANALOG OUTPUT MODULE AQ 4 X U/I ST, 16BIT RESOLUTION, 4 CHANNELS IN GROUPS OF 4 (order number: 6ES7 532-5HD00-0AB0)

You must add the following module yourself:

- 1X SIMATIC PM 190W 120/230VAC STABILIZED POWER SUPPLY Input: 120/230 VAC output: 24 V DC / 8 A (order number: 6EP1333-4BA00)

5 Planning

Because this is a new system, a new project must be created.

The hardware for this project is already specified by the existing hardware (here: SIMATIC S7-1516F PN/DP Trainer Package). Therefore, a selection does not have to be made. Instead, the listed modules of the Trainer Package are detected directly. The order numbers (see Task or Table 1) can be used for checking purposes.

Module	Order number	Slot	Address area
CPU 1516F-3 PN/DP	6ES7 516-3FN01-0AB0	1	
DI 32x24VDC HF	6ES7 521-1BL00-0AB0	2	0...3
DQ 32 X DC24V / 0.5A HF	6ES7 522-1BL01-0AB0	3	0...3
AI 8 X U/I/RTD/TC, 16BIT	6ES7 531-7KF00-0AB0	4	64...79
AQ 4 X U/I ST, 16BIT	6ES7 532-5HD00-0AB0	5	64...71

Table 1: Overview of the planned configuration

The address areas must now be configured.

The power module is not automatically detected and must be manually added.

Module	Order number	Slot	Address area
PM 190W 120/230VAC	6EP1333-4BA00	0	

Table 2: Module to be manually added

As the final step, the hardware configuration must be compiled and downloaded. Any errors present can be detected during compilation and incorrect modules can be detected when the controller is started (*only possible when hardware is present and structured identically*).

The tested project must be saved and archived.

6 Structured step-by-step instructions

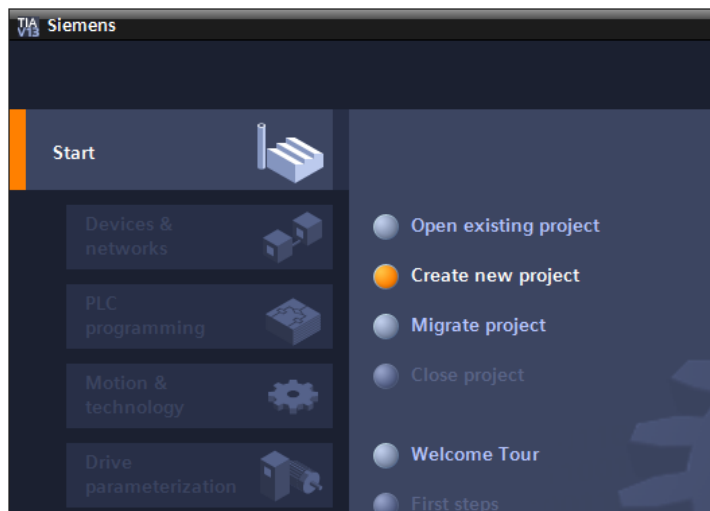
You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it is sufficient to focus on the numbered steps. Otherwise, simply follow the steps of the instructions illustrated below.

6.1 Create a new project

- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V13)



- In the portal view under the "Start" menu, select the command → "Create new project".

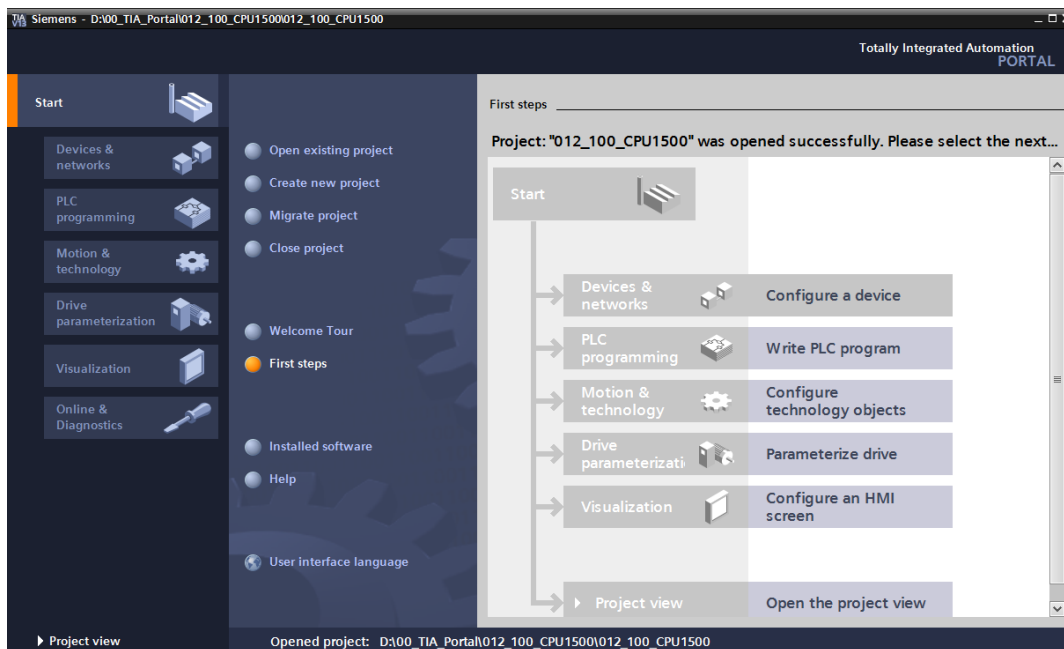


- Modify Project name, Path, Author and Comment as appropriate and click → "Create".

- The project will be created and opened and the menu "Start", "First steps" will open automatically.

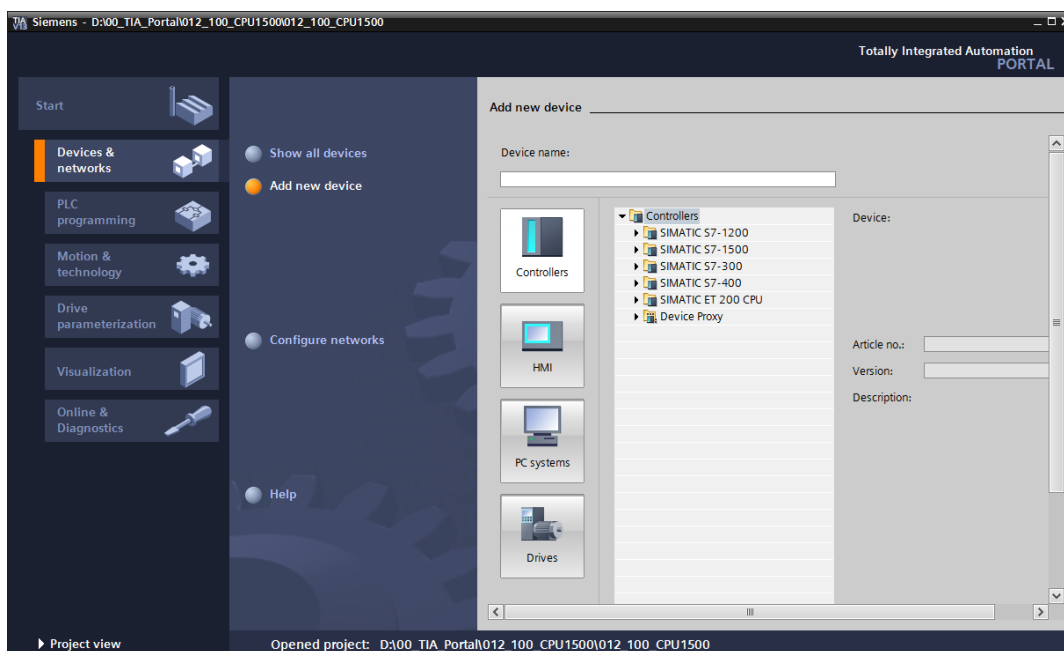
6.2 Read the hardware of the SIMATIC S7-1500

→ In the → "Start" portal, select → "First steps" → "Devices & Networks" → "Configure a device".



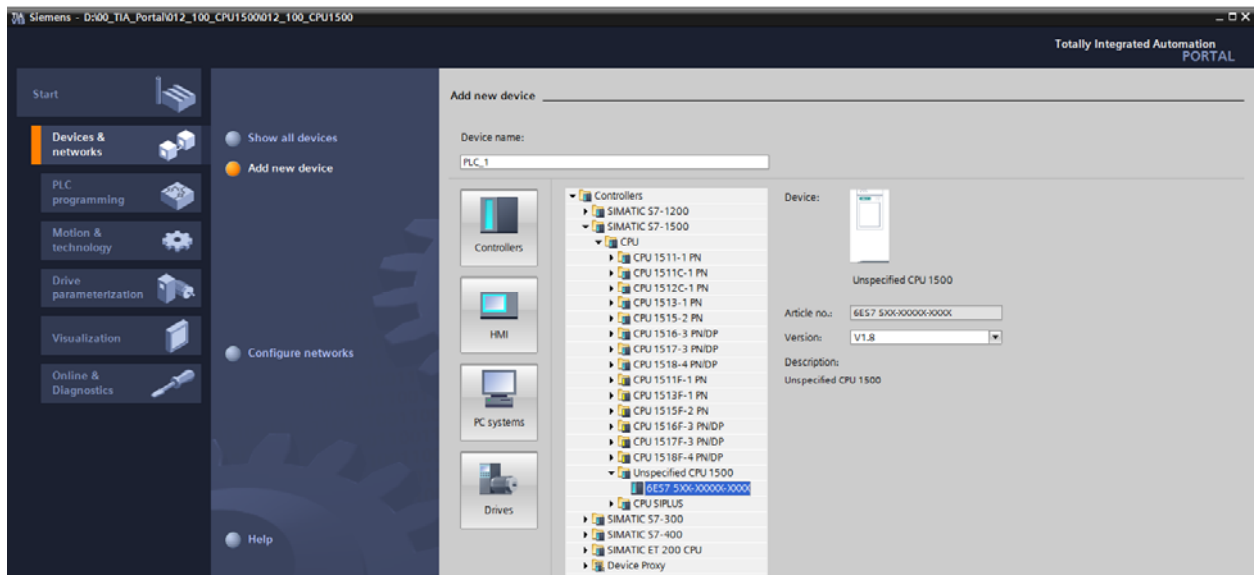
→ The "Show all devices" menu opens in the "Devices & Networks" portal.

→ Switch to the "Add new device" menu.



→ Create a new CPU. Use an unspecified model of the S7-1500 CPU with order number 6ES7 5XX-XXXX-XXXX for this.

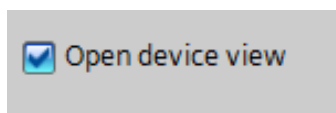
(Controllers → SIMATIC S7-1500 → CPU → Unspecified CPU 1500 → 6ES75XX-XXXX-XXXX → V1.8)



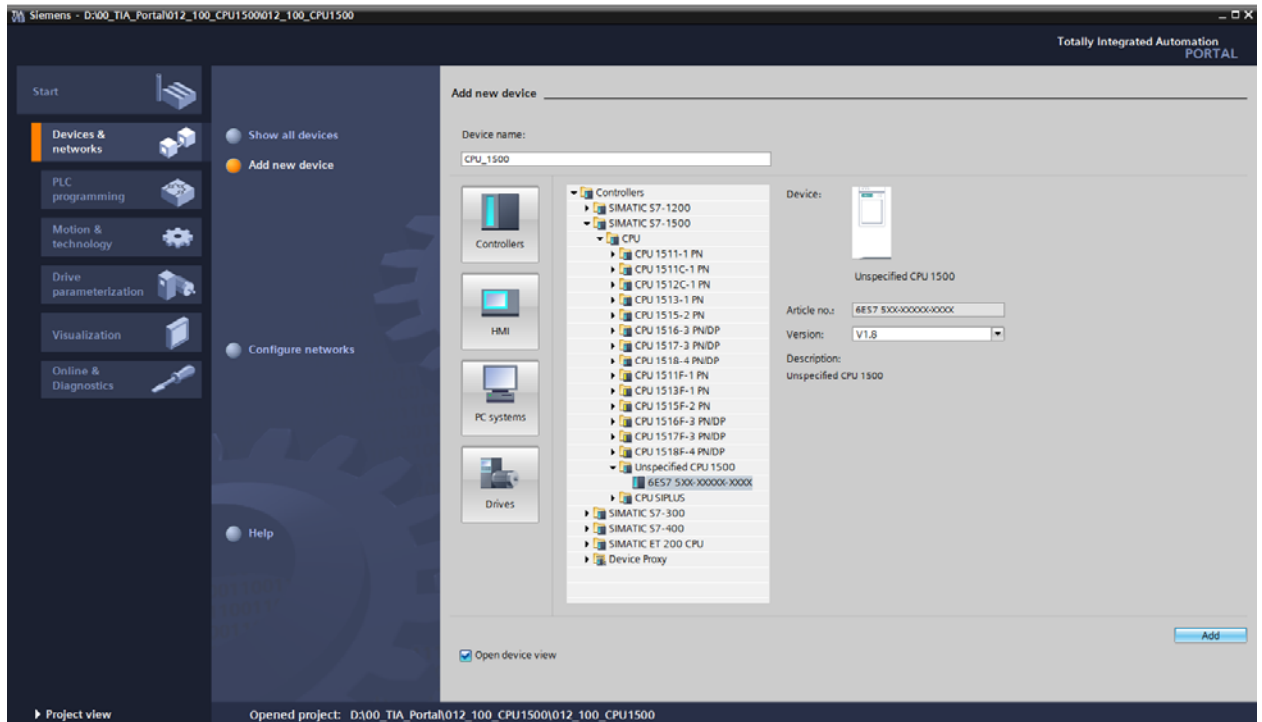
→ Assign a device name (Device name → "CPU_1500").



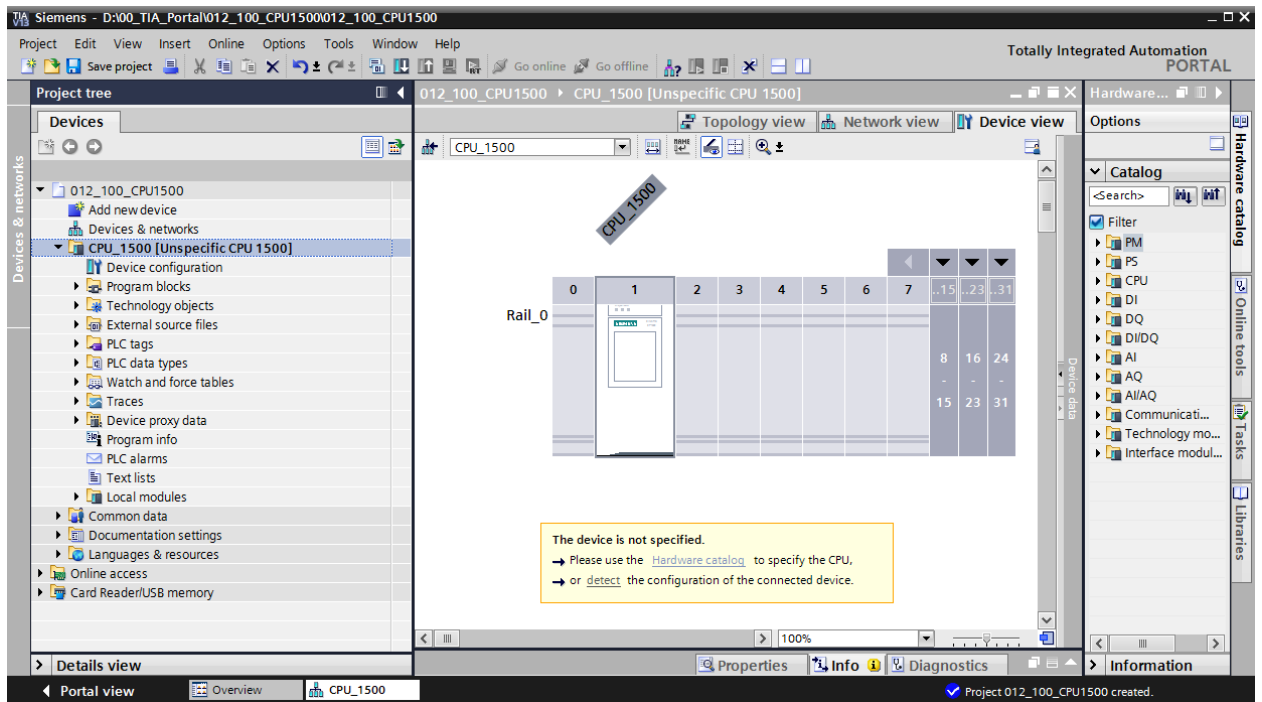
→ Select "Open device view".



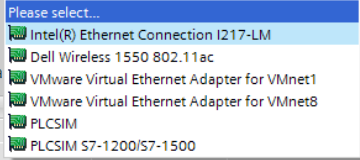
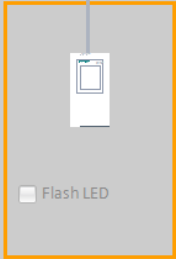
→ Click "Add".



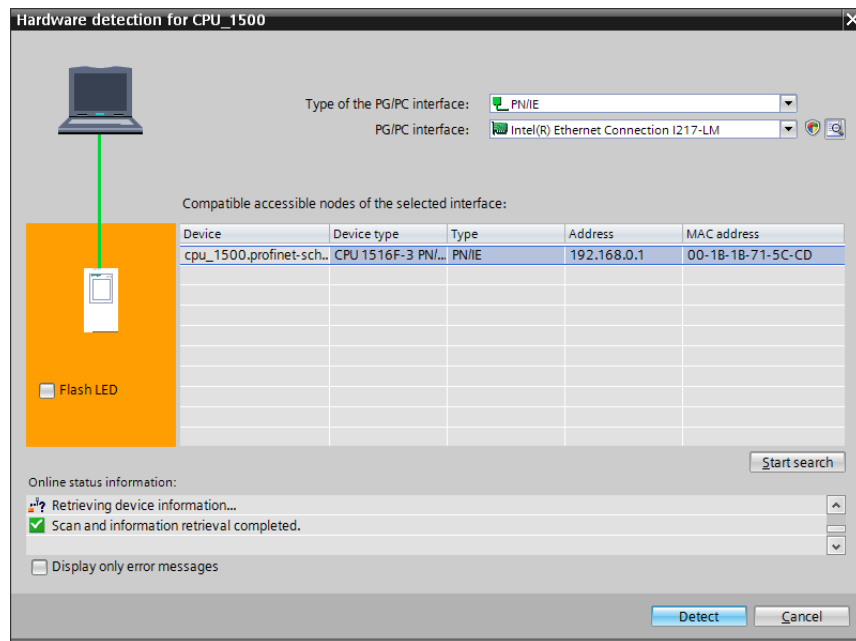
→ The TIA Portal now switches automatically to the project view and displays a notice there that this device is not specified. In order to have the hardware configuration automatically detected, start detection by clicking "detect" in the yellow information box (→ detect).



2

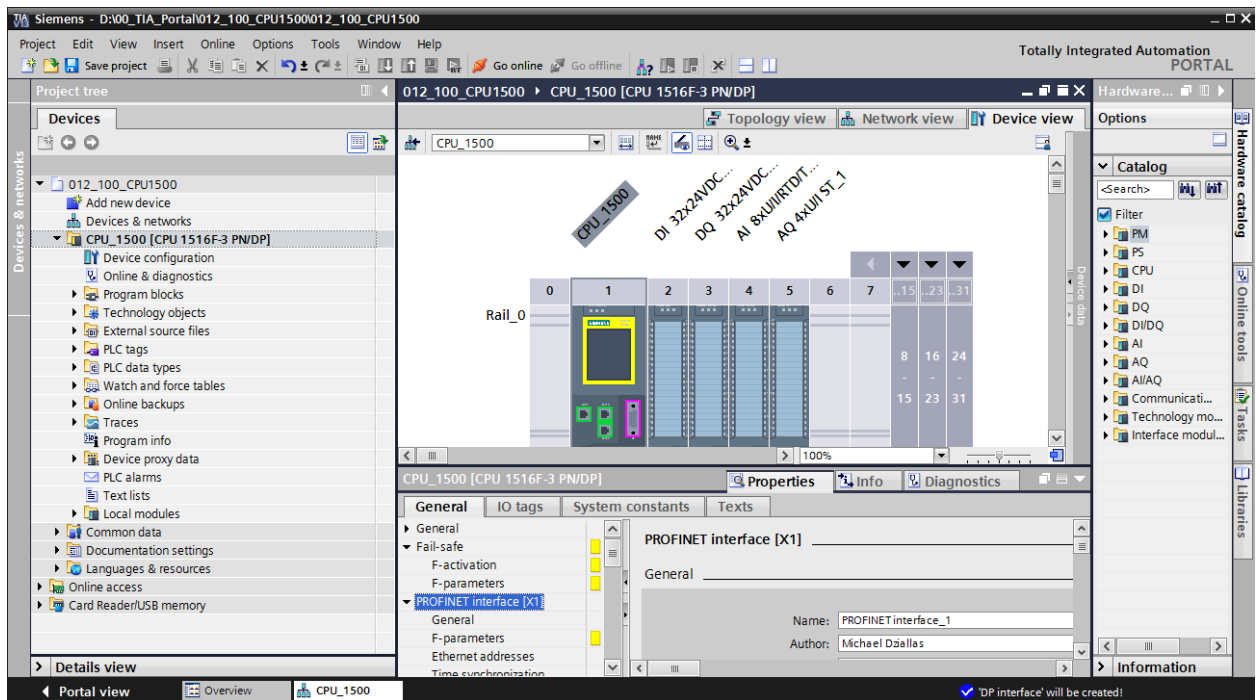


2



For unrestricted use in educational and R&D institutions. © Siemens AG 2016. All rights reserved.
SCE_EN_012-100 Unspecific Hardware Configuration S7-1500 R1512.docx

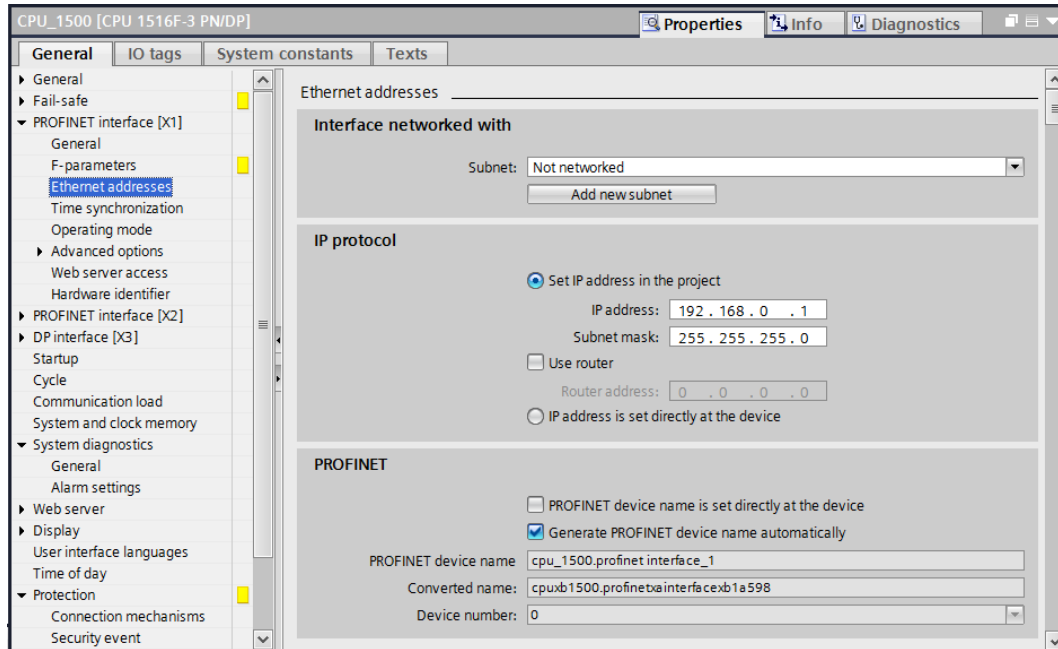
→ The TIA Portal shows the complete device configuration of the selected CPU. Only the power module is lacking. This can be placed on slot 0 of the mounting rail later.



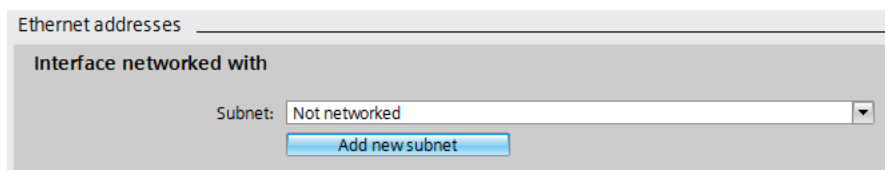
Note: You can now configure the CPU there according to your specifications. Possible settings include the PROFINET and PROFIBUS DP interfaces, startup characteristics, cycle, password protection, communication load and many others.

6.3 Configure the Ethernet interface of the CPU 1516F-3 PN/DP

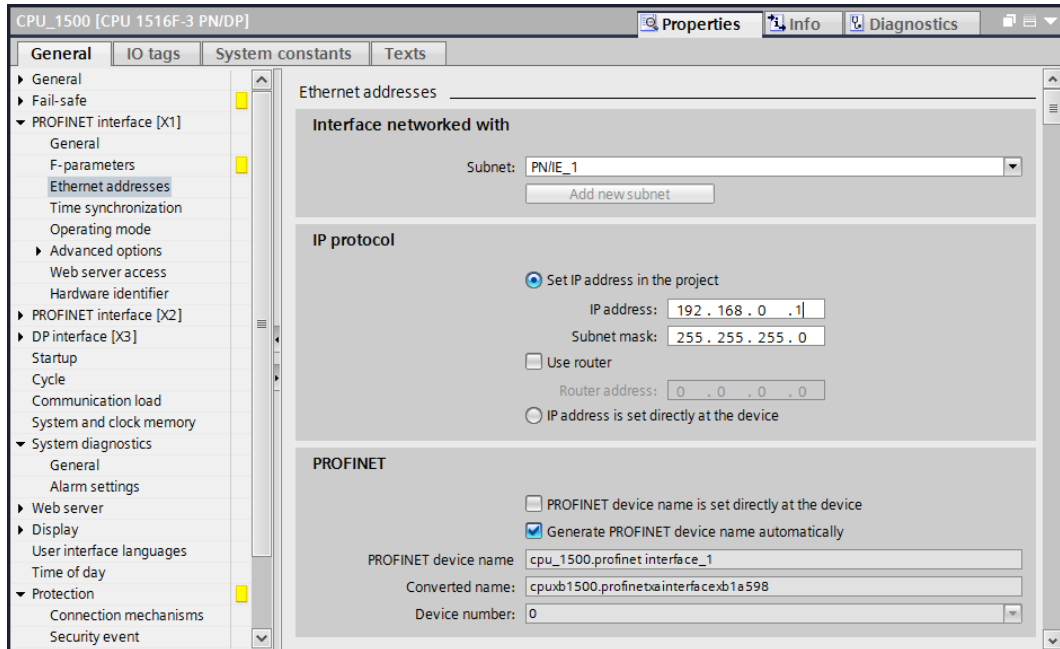
- Select the CPU with a double-click
- Under → "Properties", open the → "PROFINET-interface [X1]" menu and select the → "Ethernet addresses" entry there.



- Under "Interface connected with", only the "Not connected" entry is available.
- Add an Ethernet subnet with the → "Add new subnet" button.

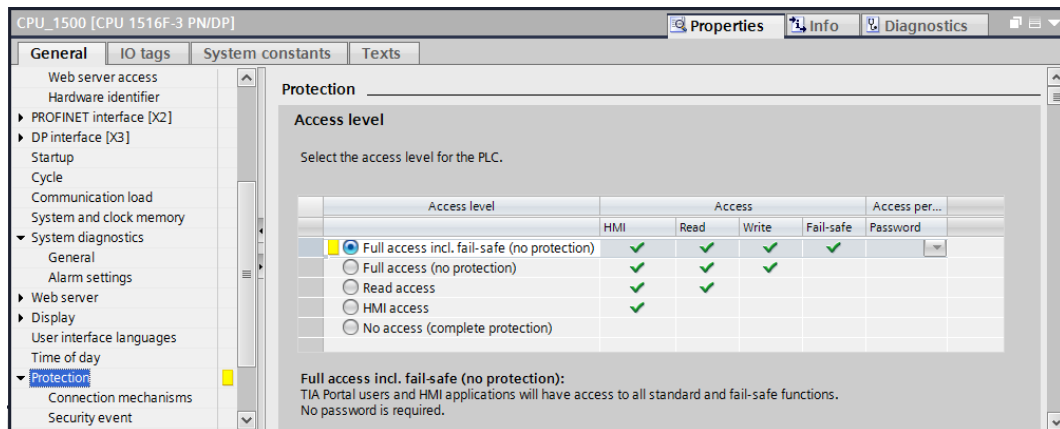


→ Keep the pre-assigned "IP address" and "Subnet mask".



6.4 Configure the access level for the CPU 1516F-3 PN/DP

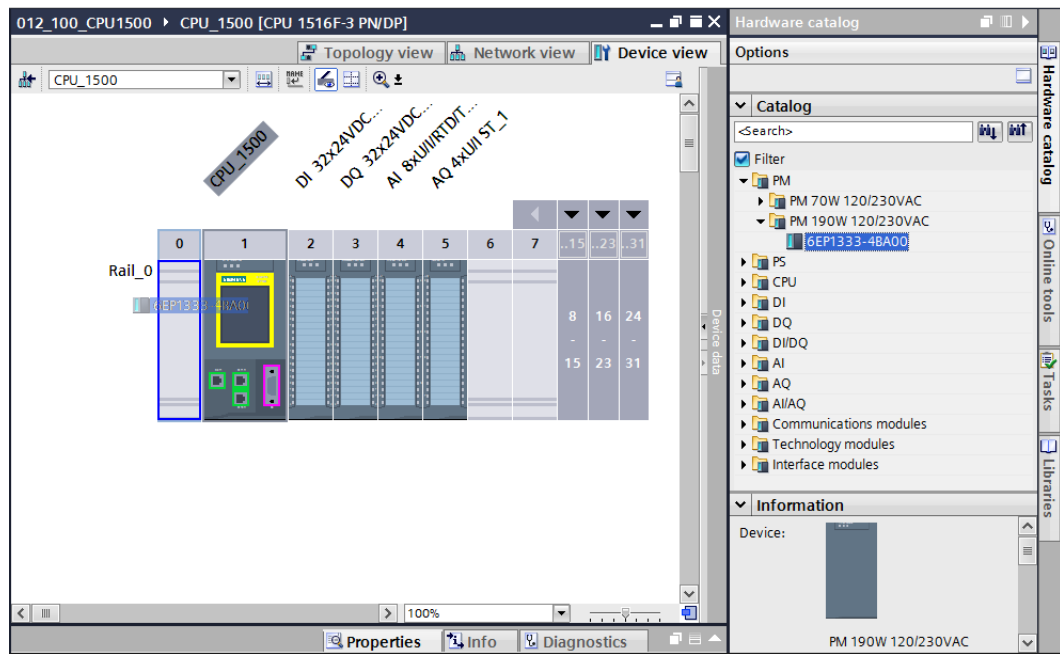
→ Switch to the → "Protection" menu and select access level → "Full access incl. fail-safe (no protection)".



Note: The setting "Full access incl. fail-safe (no protection)" is recommended because a safety program is not created here and thus we also do not have to assign a password.

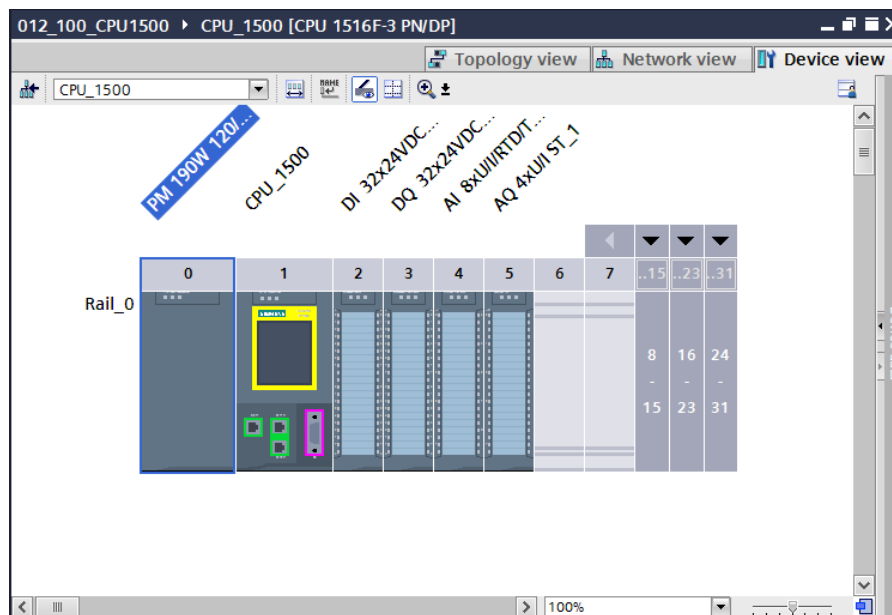
6.5 Insert power module PM 190W 120/230VAC

- Find the correct module in the hardware catalog and insert the power module into slot 0. (→ Hardware Catalog → PM → PM 190W 120/230VAC (order number 6EP1333-4BA00) → Slot 0)



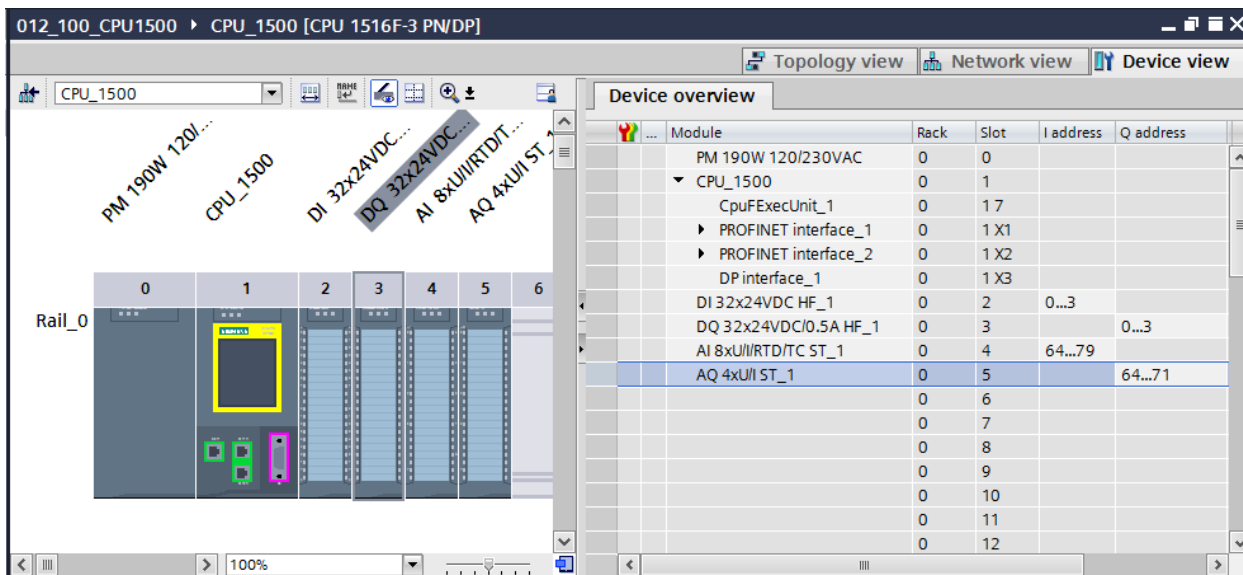
Note: If a module as well as the power module is planned for one slot, it is not possible to place it at another position even in the device configuration.

- Compare your device configuration with the following figure.



6.6 Configure the address areas of the digital input and output modules

→ The next step is to check the address areas of the inputs and output cards and adapt them if necessary. DI/DO should have an address area of 0...3 and AI/AO should have an address area of 64...79 and 64...71, respectively. (→ Device overview → DI 32x24VDC HF_1 → I address: 0...3 → DQ 32x24VDC/0.5A HF_1 → Q address: 0...3 → AI 8xU/I/RTD/TC ST_1 → I address: 64...79 → AQ 4xU/I ST_1 → Q address: 64...71)

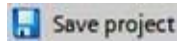


Note: To show and hide the Device overview, you must click the small arrow next to "Device data" on the right side of the hardware configuration.



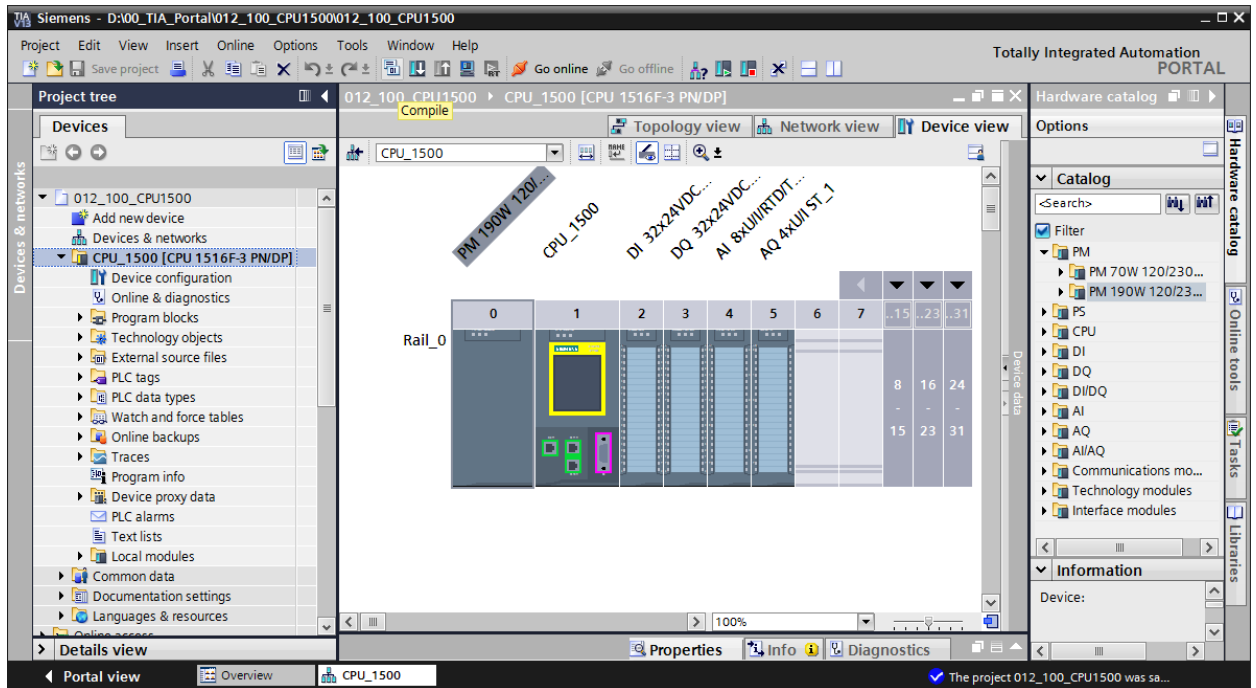
6.7 Save and compile the hardware configuration

→ Before you compile the configuration, you should save your project by clicking the →



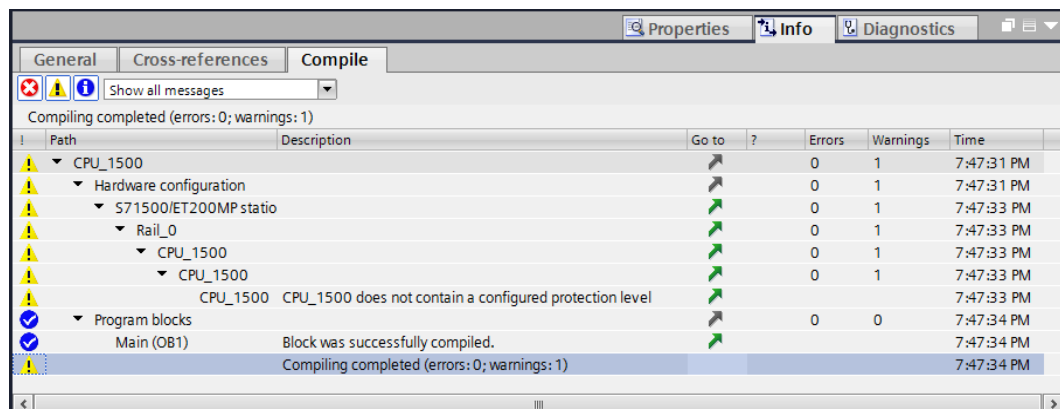
button. To compile your CPU with the device configuration, first select the →

"CPU_1500 [CPU1516F-3 PN/DP]" folder and click the → "Compile" icon.




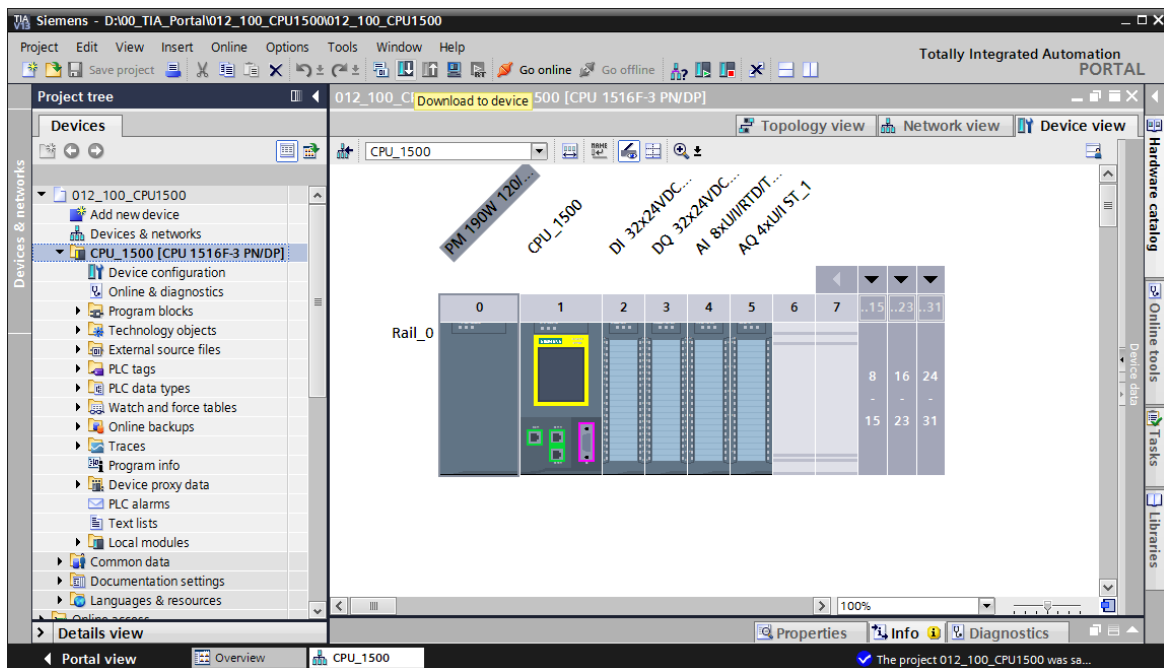
Note: "Save project" should be used repeatedly when working on a project since this does not happen automatically. A prompt to save the project only occurs when the TIA Portal is closed.

→ If the project was compiled without errors, you see the following screen.

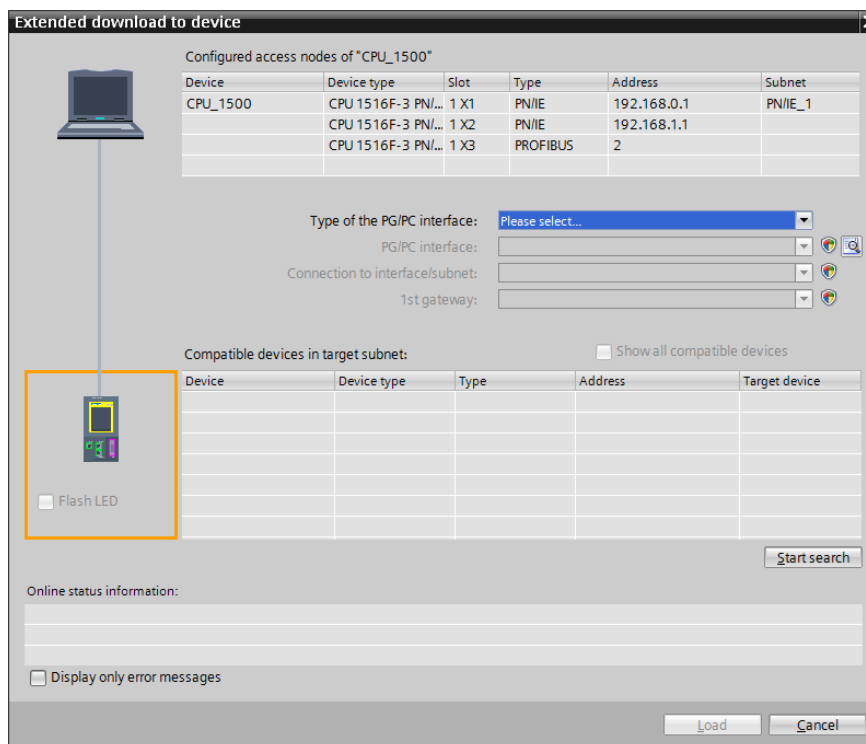


6.8 Download the hardware configuration to the device

→ To download your entire CPU, select the → "CPU_1500 [CPU1516F-3 PN/DP]" folder and click the  → "Download to device" icon.

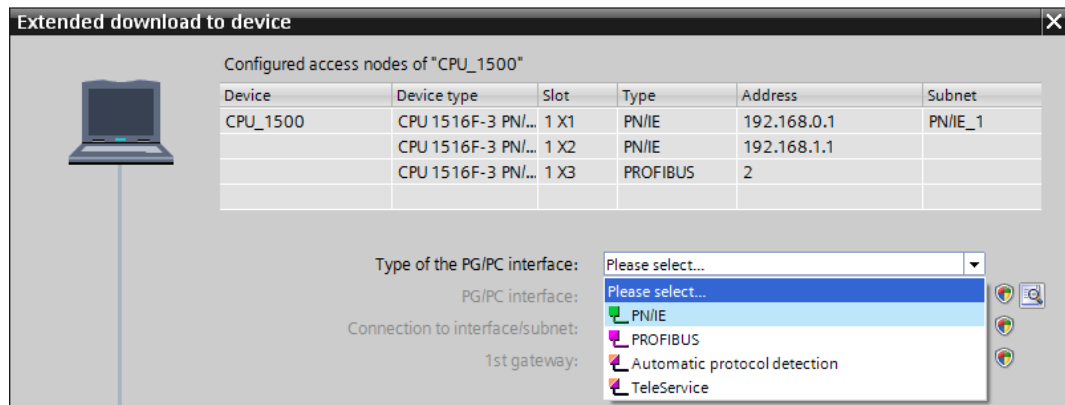


→ The manager for configuring the connection properties (extended download) opens.

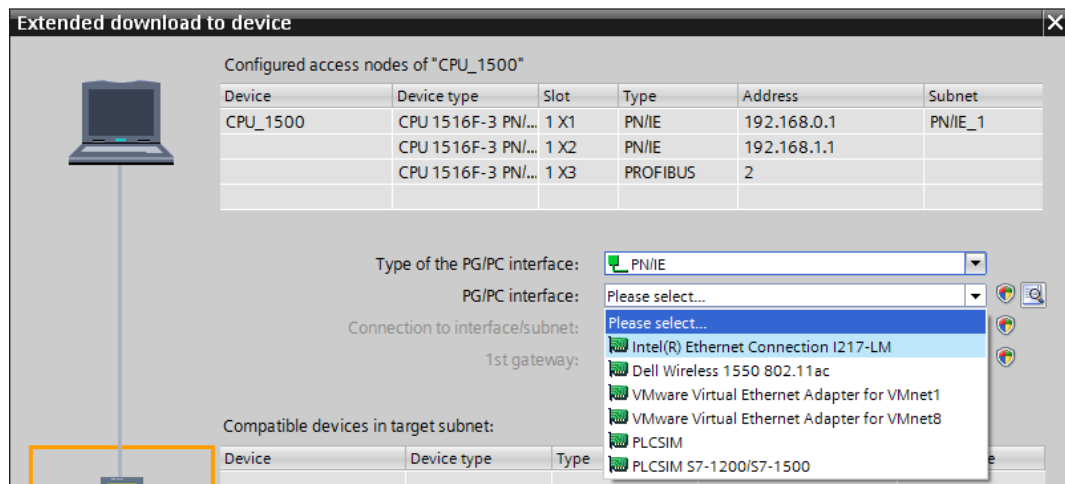


→ First, the interface must be correctly selected. This happens in three steps.

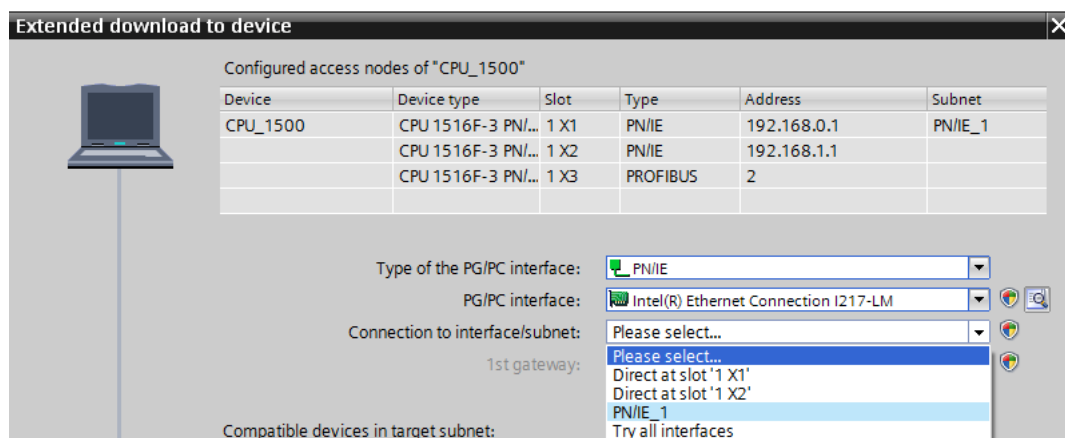
→ Type of the PG/PC interface → PN/IE



→ PG/PC interface → here: Intel(R) Ethernet Connection I217-LM

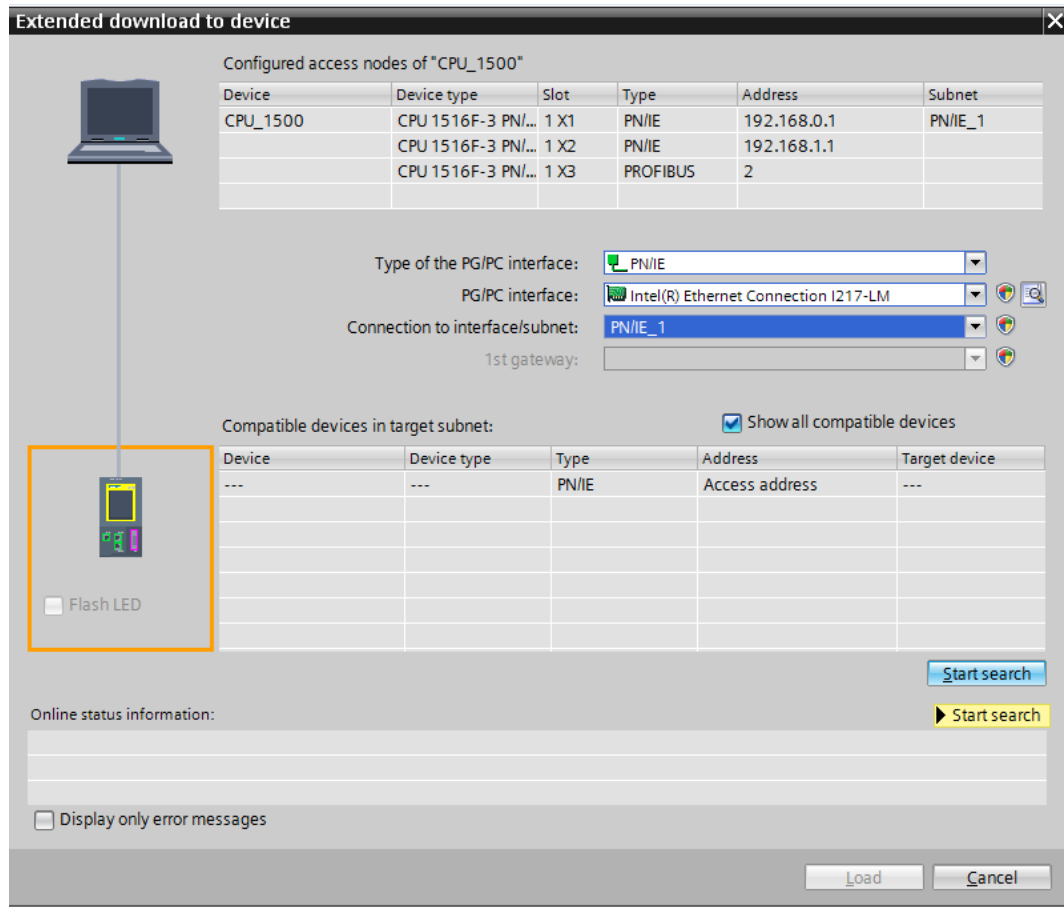


→ Connection to interface/subnet → "PN/IE_1"

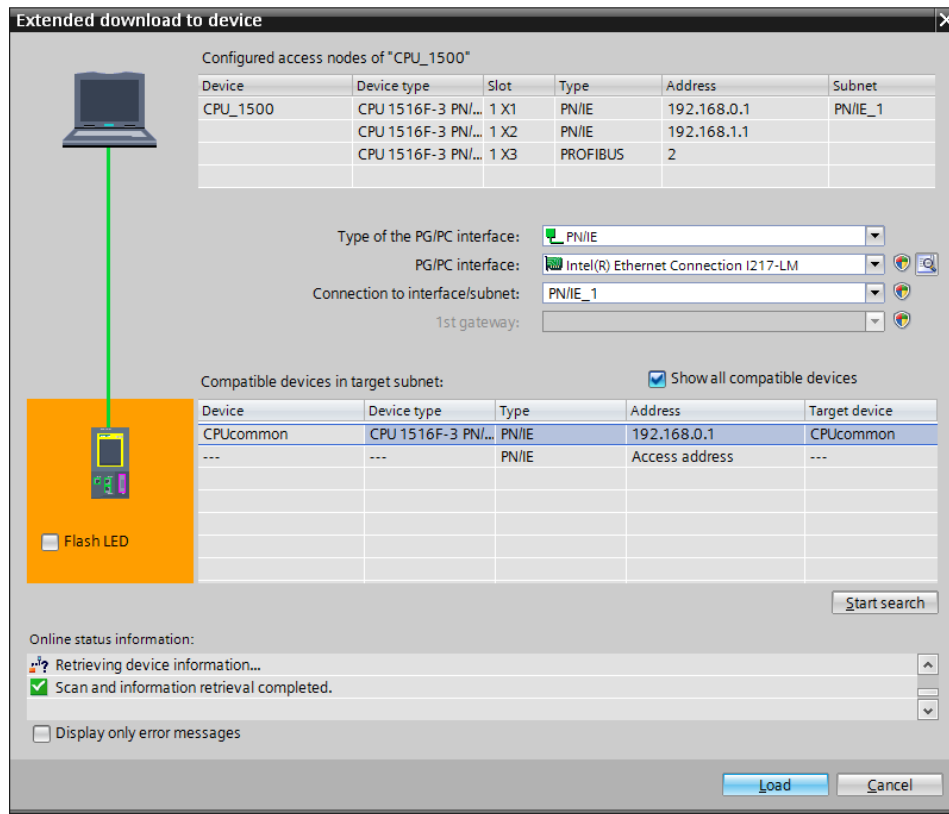


→ The → "Show all compatible devices" check box must be selected. The search for devices in the network is started by clicking the → **Start search** button.

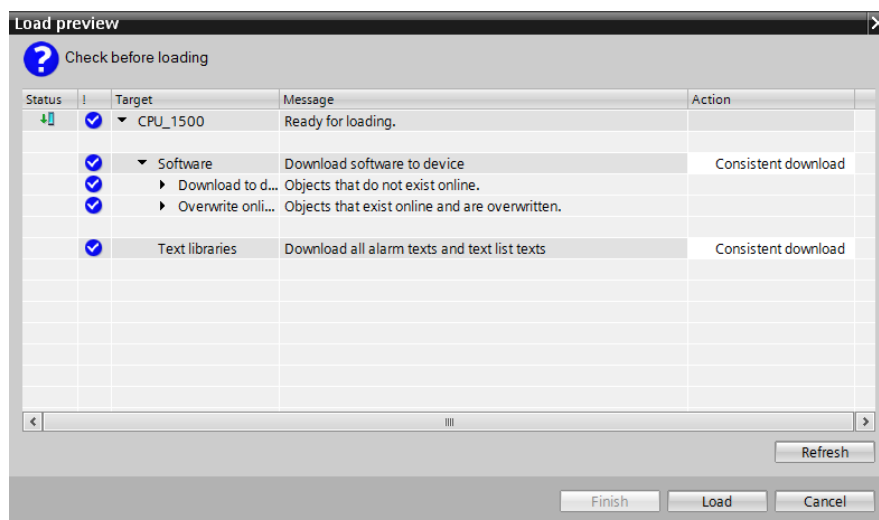
2



- If your CPU is shown in the "Compatible devices in target subnet" list, it must be selected.
The download can then be started. (→ CPU 1516F-3 PN/DP → "Load")



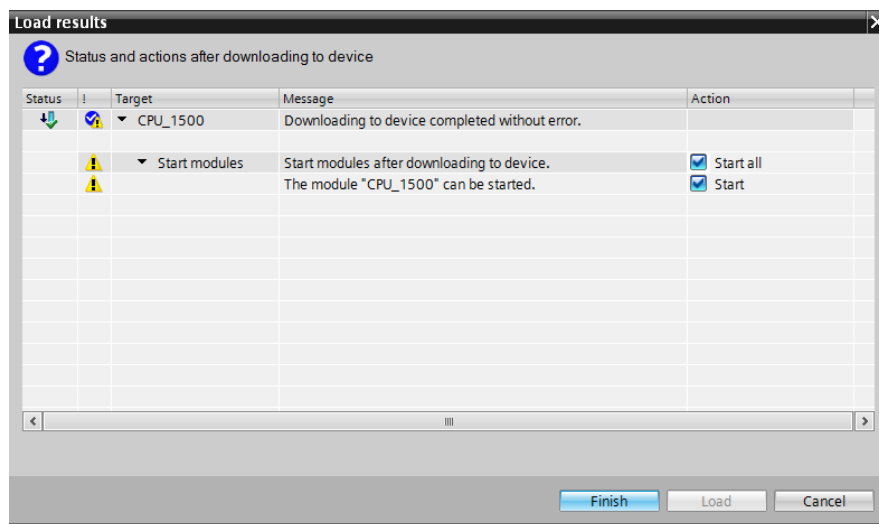
- You first obtain a preview. Confirm the prompt → "Overwrite all" and continue with → "Load".



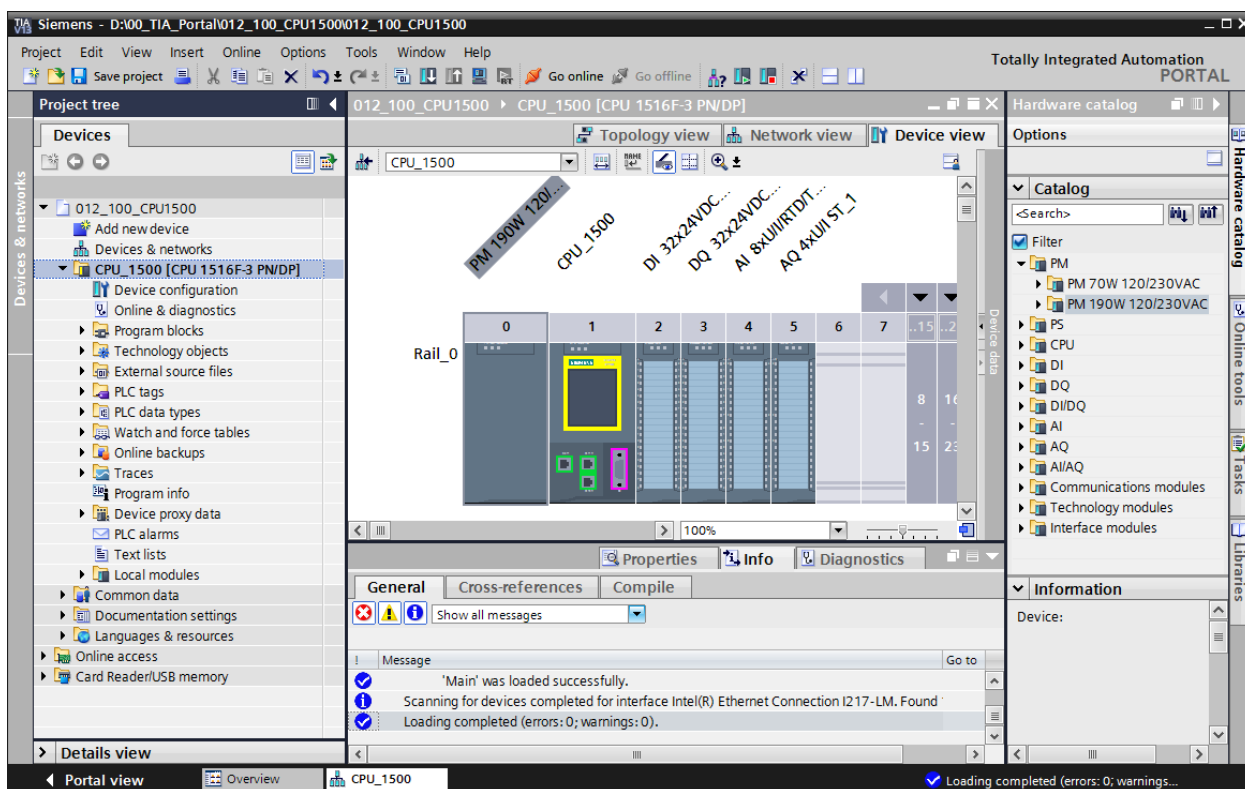
Note: The ✓ symbol should be visible in every line of the "Load preview". You can find additional information in the "Message" column.

- The → "Start all" option will be selected next before the download operation can be completed with → "Finish".

2

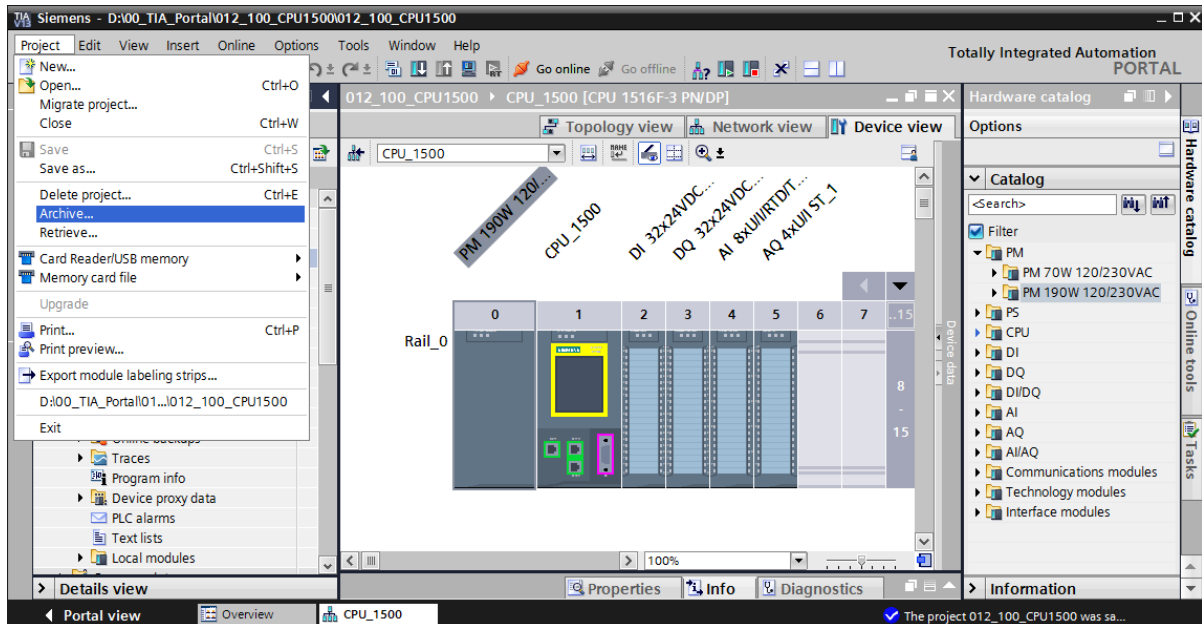


- After a successful download, the project view will open again automatically. A loading report appears in the information field under "General". This can be helpful when troubleshooting an unsuccessful download.

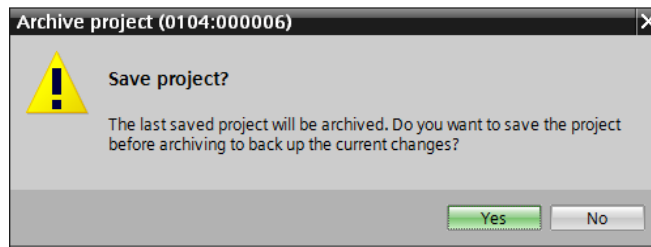


6.9 Archive the project

→ To archive the project, select the → "Archive ..." item in the → "Project" menu.



→ Confirm the prompt to save the project with → "Yes".



→ Select a folder where you want to archive your project and save it as file type "TIA Portal project archive". (→ "TIA Portal project archive" → "SCE_EN_012-100_Hardware configuration_S7-1500..." → "Save")

6.10 Checklist

No.	Description	Completed
1	Project was created	
2	Slot 0: Power module with correct order number	
3	Slot 1: CPU with correct order number	
4	Slot 1: CPU with correct firmware version	
5	Slot 2: Digital input module with correct order number	
6	Slot 2: Digital input module with correct firmware version	
7	Address area of the digital input module is correct	
8	Slot 3: Digital output module with correct order number	
9	Slot 3: Digital output module with correct firmware version	
10	Slot 3: Address area of the digital output module is correct	
11	Slot 4: Analog input module with correct order number	
12	Slot 4: Analog input module with correct firmware version	
13	Slot 4: Address area of the analog input module is correct	
14	Slot 5: Analog output module with correct order number	
15	Slot 5: Analog output module with correct firmware version	
16	Slot 5: Address area of the analog output module is correct	
17	Hardware configuration was compiled without error message	
18	Hardware configuration was downloaded without error message	
19	Project was successfully archived	

7 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

2

[illegible]

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7 CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, get in touch with your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	5
2	Prerequisite.....	5
3	Theory.....	5
3.1	SIMATIC S7-1500 automation system	5
3.1.1	Range of modules	7
3.1.2	Example configuration	10
3.2	Operator control and display elements of the CPU 1516F-3 PN/DP	11
3.2.1	Front view of the CPU 1516F-3 PN/DP with integrated display	11
3.2.2	Status and error displays.....	11
3.2.3	Operator control/connection elements of the CPU 1516F-3 PN/DP behind the front flap..	12
3.2.4	SIMATIC memory card	13
3.2.5	Mode switch.....	13
3.2.6	Display of the CPU	14
3.3	Memory areas of the CPU 1516F-3 PN/DP and the SIMATIC memory card.....	16
3.4	STEP 7 Professional V13 (TIA Portal V13) programming software	17
3.4.1	Project	18
3.4.2	Hardware configuration	18
3.4.3	Central and distributed automation structure	19
3.4.4	Planning the hardware.....	19
3.4.5	TIA Portal – Project view and portal view.....	20
3.4.6	Basic settings for the TIA Portal	22
3.4.7	Setting the IP address on the programming device	24
3.4.8	Setting the IP address in the CPU.....	27
3.4.9	Formatting the memory card in the CPU.....	30
3.4.10	Resetting the CPU to factory settings.....	31
4	Task	32

5	Planning.....	32
6	Structured step-by-step instructions.....	33
6.1	Create a new project.....	33
6.2	Insert the CPU 1516F-3 PN/DP.....	34
6.3	Configure the Ethernet interface of the CPU 1516F-3 PN/DP	38
6.4	Configure the fail-safe operation of the CPU 1516F-3 PN/DP	39
6.5	Configure the access level for the CPU 1516F-3 PN/DP	40
6.6	Insert the digital input module DI 32x24VDC HF.....	40
6.7	Insert the digital output module DQ 32xDC24V / 0.5A HF	42
6.8	Insert power module PM 190W 120/230VAC.....	43
6.9	Configure the address areas of the digital input and output modules	44
6.10	Save and compile the hardware configuration	45
6.11	Download the hardware configuration to the device	46
6.12	Download the hardware configuration to the PLCSIM simulation (optional).....	51
6.13	Archive the project.....	59
6.14	Checklist	60
7	Exercise	61
7.1	Task – Exercise	61
7.2	Planning.....	61
7.3	Checklist – Exercise	62
8	Additional information	63

SPECIFIED HARDWARE CONFIGURATION – SIMATIC S7-1516F PN/DP

1 Goal

In this chapter, you will first learn how to ***create a project***. You are then shown how the ***hardware is configured***.

2 Prerequisite

You do not need any previous knowledge from other chapters to successfully complete this chapter.

3 Theory

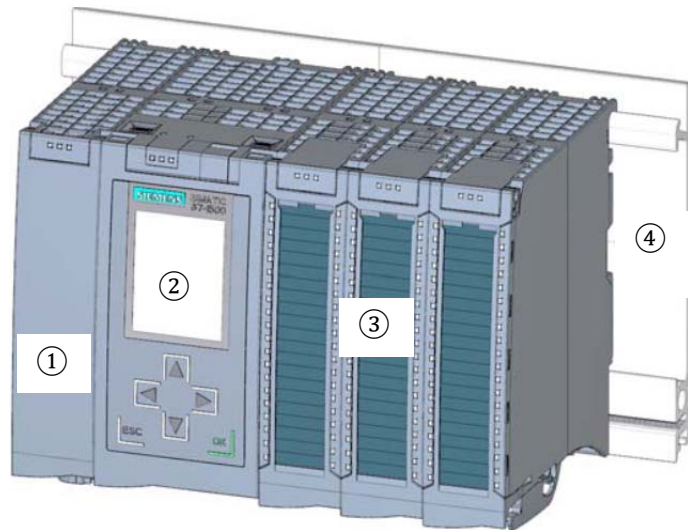
3.1 SIMATIC S7-1500 automation system

The SIMATIC S7-1500 automation system is a modular controller system for the middle to upper performance range. A comprehensive range of modules is available to optimally adapt the system to the automation task.

SIMATIC S7-1500 is the next generation of the SIMATIC S7-300 and S7-400 automation systems with the following new performance features.

- Increased system performance
- Integrated motion control functionality
- PROFINET IO IRT
- Integrated display for machine-level operation and diagnostics
- STEP 7 language innovations while maintaining proven functions

The S7-1500 controller consists of a power supply ①, a CPU with integrated display ② and input and output modules for digital and analog signals ③. The modules are mounted on a mounting rail with integrated DIN rail profile ④. If necessary, communication processors and function modules are also used for special tasks such as stepper motor control.



The programmable logic controller (PLC) uses the S7 program to monitor and control a machine or process. In doing so, the S7 program scans the I/O modules via input addresses (%I) and addresses their output addresses (%Q).

The system is programmed with the STEP 7 Professional V13 software.

3.1.1 Range of modules

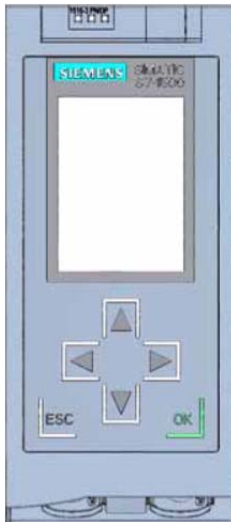
The SIMATIC S7-1500 is a modular automation system and offers the following range of modules:

Central processing units (CPUs) with integrated display

The CPUs have different performance capability and execute the user program. In addition, the other modules are supplied power via the backplane bus with the integrated system power supply.

Additional properties and functions of the CPU:

- Communication via Ethernet
- Communication via PROFIBUS/PROFINET
- HMI communication for HMI devices
- Web server
- Integrated technology functions (e.g. PID controller, motion control, etc.)
- System diagnostics
- Integrated security (e.g. know-how, copy, access, integrity protection)



System power supply modules (PS) (rated input voltages 24 V DC to 230 V AC/DC)

with connection to the backplane bus supply the configured modules with the internal supply voltage.

**Load current supply modules (PM) (rated input voltages 120/230 V AC)**

do not have a connection to the backplane bus of the S7-1500 automation system. The load current supply is used to supply 24 V DC to the system power supply of the CPU, the input and output circuits of I/O modules and the sensors and actuators.



I/O modules

for digital input (DI) / digital output (DQ) / analog input (AI) / analog output (AQ)

**Technology modules (TM)**

as incremental encoders and pulse encoders with/without direction signal.

**Communication modules (CM)**

for serial communication RS232 / RS422 / RS485, PROFIBUS and PROFINET.



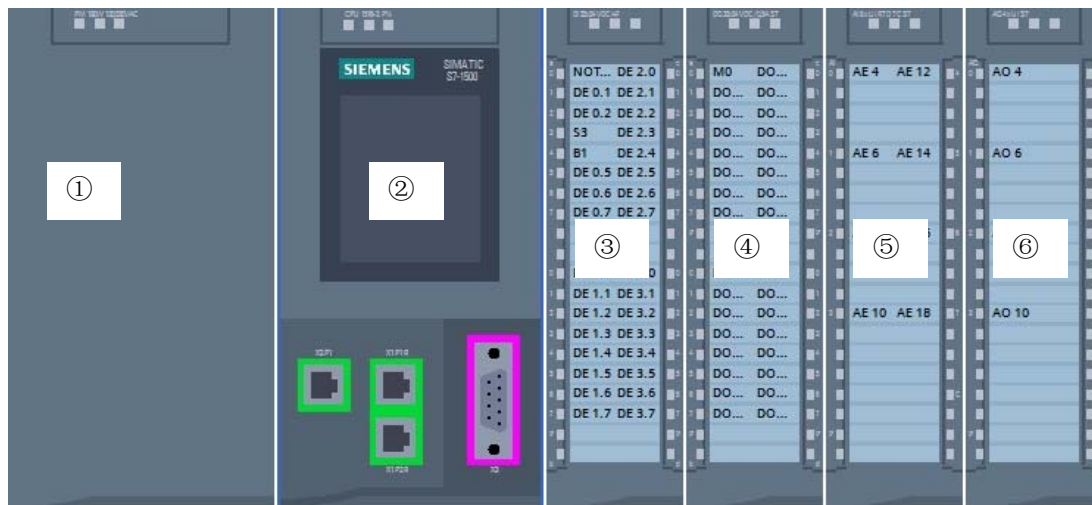
SIMATIC memory card

up to a maximum of 2 GB for storing program data and for easy replacement of CPUs during maintenance.



3.1.2 Example configuration

The following configuration of an S7-1500 automation system will be used for the program example in this curriculum.



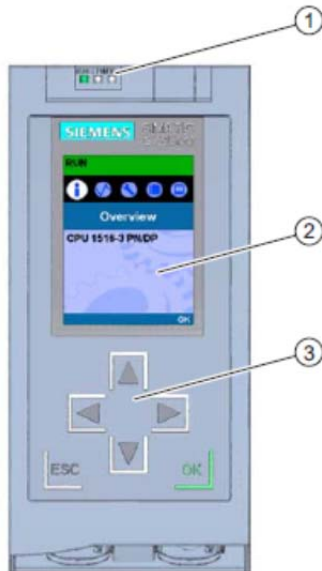
- ① Load current supply module (PM) with 120/230 V AC, 50 Hz / 60 Hz, 190 W input and 24 V DC / 8 A output
- ② Central processing unit CPU 1516F-3 PN/DP with integrated PROFIBUS and PROFINET interfaces
- ③ I/O module 32x digital input DI 32x24VDC HF
- ④ I/O module 32x digital output DQ 32x24VDC/0.5A HF
- ⑤ I/O module 8x analog input AI 8xU/I/RTD/TC ST
- ⑥ I/O module 4x analog output AQ 4xU/I ST

3.2 Operator control and display elements of the CPU 1516F-3 PN/DP

The figure below shows the operator control and display elements of a CPU 1516F-3 PN/DP.

The arrangement and number of elements differ from this figure for other CPUs.

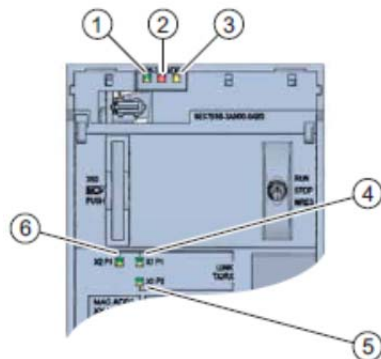
3.2.1 Front view of the CPU 1516F-3 PN/DP with integrated display



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Display
- ③ Control keys

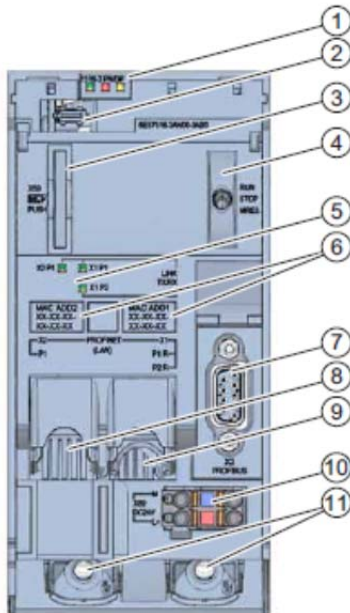
3.2.2 Status and error displays

The CPU comes with the following LED displays:



- ① RUN/STOP LED (yellow/green LED)
- ② ERROR LED (red LED)
- ③ MAINT LED (yellow LED)
- ④ LINK RX/TX LED for port X1 P1 (yellow/green LED)
- ⑤ LINK RX/TX LED for port X1 P2 (yellow/green LED)
- ⑥ LINK RX/TX LED for port X2 P1 (yellow/green LED)

3.2.3 Operator control/connection elements of the CPU 1516F-3 PN/DP behind the front flap



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Display connection
- ③ Slot for the SIMATIC memory card
- ④ Mode switch
- ⑤ LED displays for the 3 ports of the PROFINET interfaces X1 and X2
- ⑥ MAC addresses of the interfaces
- ⑦ PROFIBUS interface (X3)
- ⑧ PROFINET interface (X2) with 1 port
- ⑨ PROFINET interface (X1) with 2-port switch
- ⑩ Connection for supply voltage
- ⑪ Fastening screws

Note: The front flap with the display can be removed and inserted during operation.

3.2.4 SIMATIC memory card

A SIMATIC Micro Memory Card is used as the memory module for the CPUs. This is a preformatted memory card that is compatible with the Windows file system. It is available with various storage capacities and can be used for the following purposes:

- Transportable data storage medium
- Program card
- Firmware update card

The MMC **must** be inserted to operate the CPU as the CPUs have no integrated load memory. A commercially available SD card reader is needed to write/read the SIMATIC memory card with the programming device or PC. This allows files to be copied directly to the SIMATIC memory card using Windows Explorer, for example.

Note: It is recommended that the SIMATIC memory card only be removed or inserted when the CPU is in the POWER OFF state.

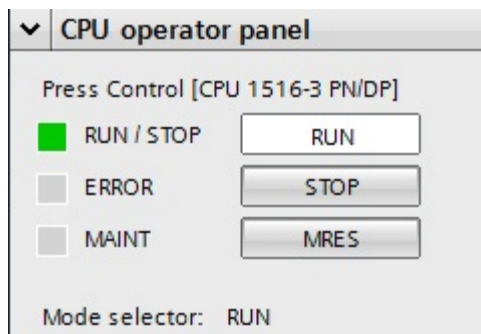
3.2.5 Mode switch

The mode switch allows you to set the operating mode of the CPU. The mode switch is designed as a toggle switch with 3 switch positions.

Position	Meaning	Explanation
RUN	RUN mode	The CPU processes the user program.
STOP	STOP mode	The CPU is not executing the user program.
MRES	Memory reset	Position for CPU memory reset.

You can also use the button on the CPU operator panel of the STEP 7 Professional V13 software in Online & Diagnostics to switch the operating mode (**STOP** or **RUN**).

The operator panel also contains an **MRES** button for performing a memory reset and displays the status LEDs of the CPU.



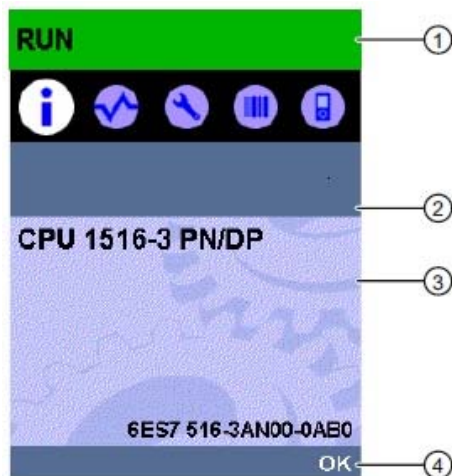
3.2.6 Display of the CPU

The S7-1500 CPU has a front flap with a display and control keys. Control data and status data can be displayed in various menus on the display and numerous settings can be configured. You use the control keys to navigate through the menus.

The display of the CPU offers the following functions:

- 6 different display languages can be selected.
- Diagnostic messages are displayed in plain text.
- The interface settings can be changed locally.
- Password assignment for display operation is possible through the TIA Portal.

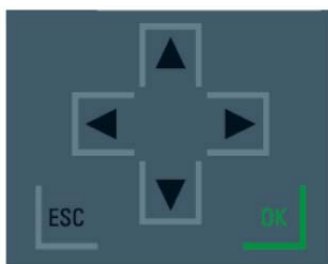
View of the display of an S7-1500:



- ① CPU status information
- ② Submenu name
- ③ Information display field
- ④ Navigation aid. e.g. OK/ESC or the page number

Control keys of the display






- Four arrow keys: "up", "down", "left", "right"
- An ESC key
- An OK key



Functions of the "OK" and "ESC" keys

- For menu commands in which an input can be made:
 - OK → valid access to the menu command, confirmation of input and exit from editing mode
 - ESC → restoration of original content (which means changes are not saved) and exit from editing mode
- For menu commands in which no input can be made:
 - OK → to next submenu command
 - ESC → back to previous menu command

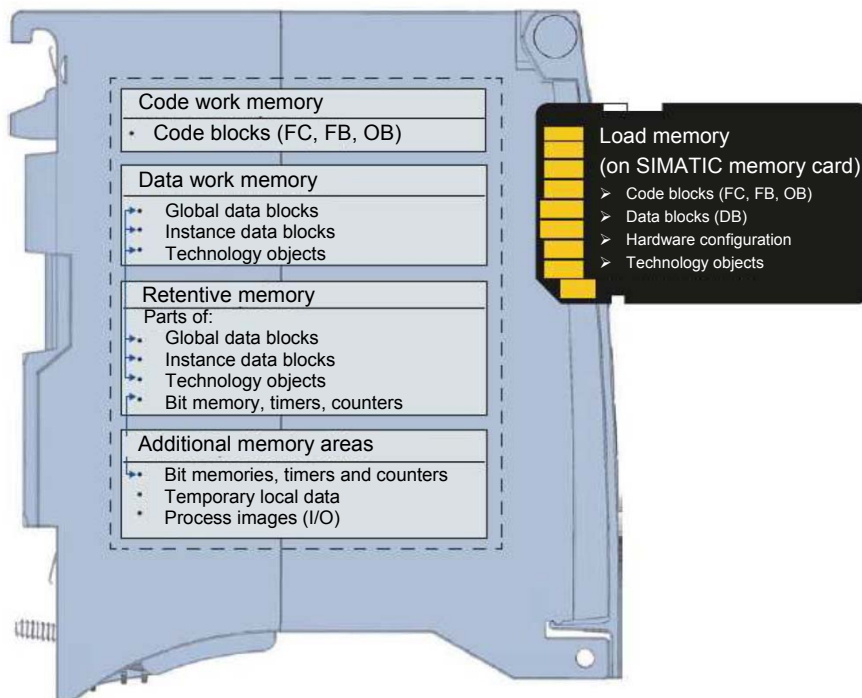
Available submenus of the display:

Main menu commands	Meaning	Explanation
	Overview	The "Overview" menu contains information about the properties of the CPU.
	Diagnostics	The "Diagnostics" menu contains information about diagnostic messages, the diagnostic description and the indication of interrupts. There is also information about the network properties of each interface of the CPU.
	Settings	In the "Settings" menu, the IP addresses of the CPU are assigned, the date, time, time zones, operating modes (RUN/STOP) and protection levels are set, the CPU memory is reset and its factory settings are restored and the status of firmware updates is displayed.
	Modules	The "Modules" menu contains information about the modules that are used in your configuration. The modules can be used as central or distributed modules. Distributed modules are connected to the CPU via PROFINET and/or PROFIBUS. You have the option here to set the IP addresses for a CPU.
	Display	In the "Display" menu, settings are made for all aspects of the display, such as the language setting, brightness setting and Energy-saving mode. (Energy-saving darkens the display. Standby mode switches off the display.)

3.3 Memory areas of the CPU 1516F-3 PN/DP and the SIMATIC memory card

The following figure shows the memory areas of the CPU and the load memory on the SIMATIC memory card.

In addition to the load memory, other data can be loaded onto the SIMATIC memory card using Windows Explorer. This includes recipes, data logs, project backups and additional documentation for the program.



Load memory

Load memory is non-volatile memory for code blocks, data blocks, technology objects and the hardware configuration. When these objects are downloaded to the CPU, they are first stored in the load memory. This memory is located on the SIMATIC memory card.

Work memory

Work memory is volatile memory that contains the code and data blocks. The work memory is integrated into the CPU and cannot be expanded. In S7-1500 CPUs, the work memory is divided into two areas:

- Code work memory:
 - The code work memory contains runtime-relevant parts of the program code.
- Data work memory:
 - The data work memory contains the runtime-relevant parts of the data blocks and technology objects.

At the operating mode transitions from POWER ON to startup and from STOP to startup, tags of global data blocks, instance data blocks and technology objects are initialized with their start values. Retentive tags retain their actual values that were saved in the retentive memory.

Retentive memory

Retentive memory is non-volatile memory for saving certain data in the event of power failure. The tags and operand areas that have been defined as retentive are saved in the retentive memory. This data is retained beyond power-off or power failure.

All other program tags are set to their start values at the operating mode transitions from POWER ON to startup and from STOP to startup.

The content of retentive memory is deleted by the following actions:

- Memory reset
- Reset to factory settings

Note: Certain tags of technology objects are also stored in the retentive memory. These tags are not deleted by a memory reset.

3.4 STEP 7 Professional V13 (TIA Portal V13) programming software

STEP 7 Professional V13 (TIA Portal V13) software is the programming tool for the following automation systems:

- SIMATIC S7-1500
- SIMATIC S7-1200
- SIMATIC S7-300
- SIMATIC S7-400
- SIMATIC WinAC

STEP 7 Professional V13 provides the following functions for plant automation:

- Configuration and parameter assignment of the hardware
- Specification of the communication
- Programming
- Testing, commissioning and servicing with operational/diagnostic functions
- Documentation
- Creation of visualizations for SIMATIC Basic Panels using the integrated WinCC Basic software
- Visualization solutions for PCs and other panels can also be created with other WinCC software packages

Support is provided for all functions through detailed online help.

3.4.1 Project

To implement a solution for an automation and visualization task, you create a project in the TIA Portal. A project in the TIA Portal contains the configuration data for the configuration and internetworking of devices as well as the programs and the configuration of the visualization.

3.4.2 Hardware configuration

The *hardware configuration* includes the configuration of the devices, consisting of the hardware of the automation system, the intelligent field devices and the hardware for visualization. The configuration of the networks specifies the communication between the various hardware components. The individual hardware components are *inserted in the hardware configuration* from catalogs.

The hardware of automation systems comprises controllers (CPUs), signal modules for input and output signals (SMs) and communication processors, and interface modules (CP, IM). Power supply and voltage supply modules (PS, PM) are also available to supply the modules.

The signal modules and intelligent field devices connect the input and output data of the process to be automated and visualized to the automation system.

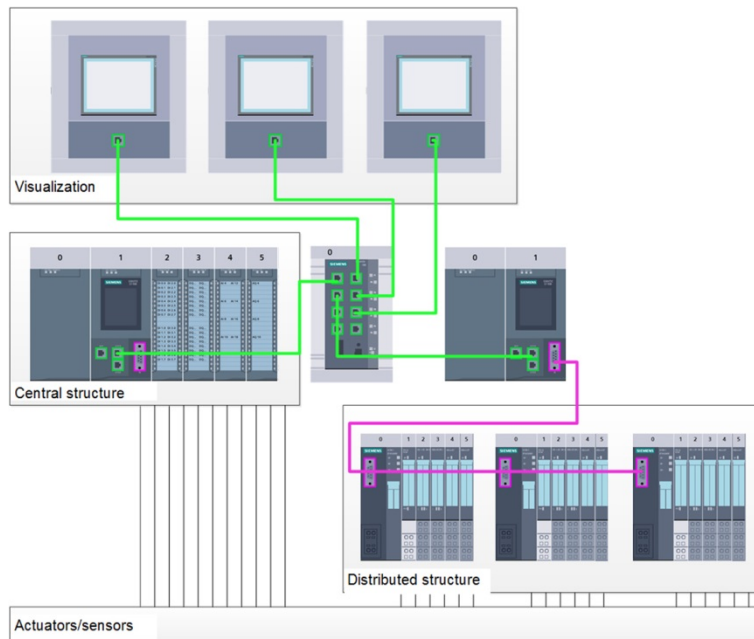


Figure 1: Example of hardware configuration with central and distributed structures

The hardware configuration enables the downloading of automation and visualization solutions to the automation system and access to the connected signal modules by the controller.

3.4.3 Central and distributed automation structure

Figure 1 shows an automation structure that contains both central and distributed structures.

In central structures, the input and output signals of the process are transmitted by way of conventional wiring to the signal modules, which are connected directly to the controller. Conventional wiring refers to the connection of sensors and actuators using 2-wire or 4-wire cables.

The distributed structure is the predominant structure used today. Here, the sensors and actuators are wired conventionally only as far as the signal modules of the field devices. The signal transmission from the field devices to the controller is implemented using an industrial communication system.

Both classic fieldbuses such as PROFIBUS, Modbus and Foundation Fieldbus as well as Ethernet-based communication systems such as PROFINET can be used as the industrial communication system.

In addition, intelligent field devices in which stand-alone programs run can also be connected via the communication system. These programs can also be created with the TIA Portal.

3.4.4 Planning the hardware

Before you can configure the hardware, you must plan it (hardware planning). In general, you begin by selecting which controllers are needed and how many. Next you select the communication modules and signal modules. The selection of signal modules is based on the number and type of inputs and outputs needed. As the final step, a power supply that ensures that the necessary power is supplied must be selected for each controller or field device.

The functionality required and the ambient conditions are of vital importance for planning the hardware configuration. For example, the temperature range in the application area sometimes limits the devices available for selection. Fail-safe operation might be another requirement, for example.

The [TIA Selection Tool](#) (Select automation technology → TIA Selection Tool and follow the instructions) provides you support. Note: TIA Selection Tool requires Java.

Note for online research: *If more than one manual is available, you should look for the description "Device Manual", "Product Manual" or simply "Manual" (as opposed to "Function Manual", "List Manual", "System Manual", etc.) in order to find the device specifications.*

3.4.5 TIA Portal – Project view and portal view

The TIA Portal has two important views. When started, the portal view appears by default. This view makes getting started easier, especially for beginning users.

The portal view provides a task-oriented view of the tools for working on the project. Here, you can quickly decide what you want to do and open the tool for the task at hand. If necessary, a change to the project view takes place automatically for the selected task.

Figure 2 shows the portal view. At the bottom left, there is an option to switch between this view and the project view.

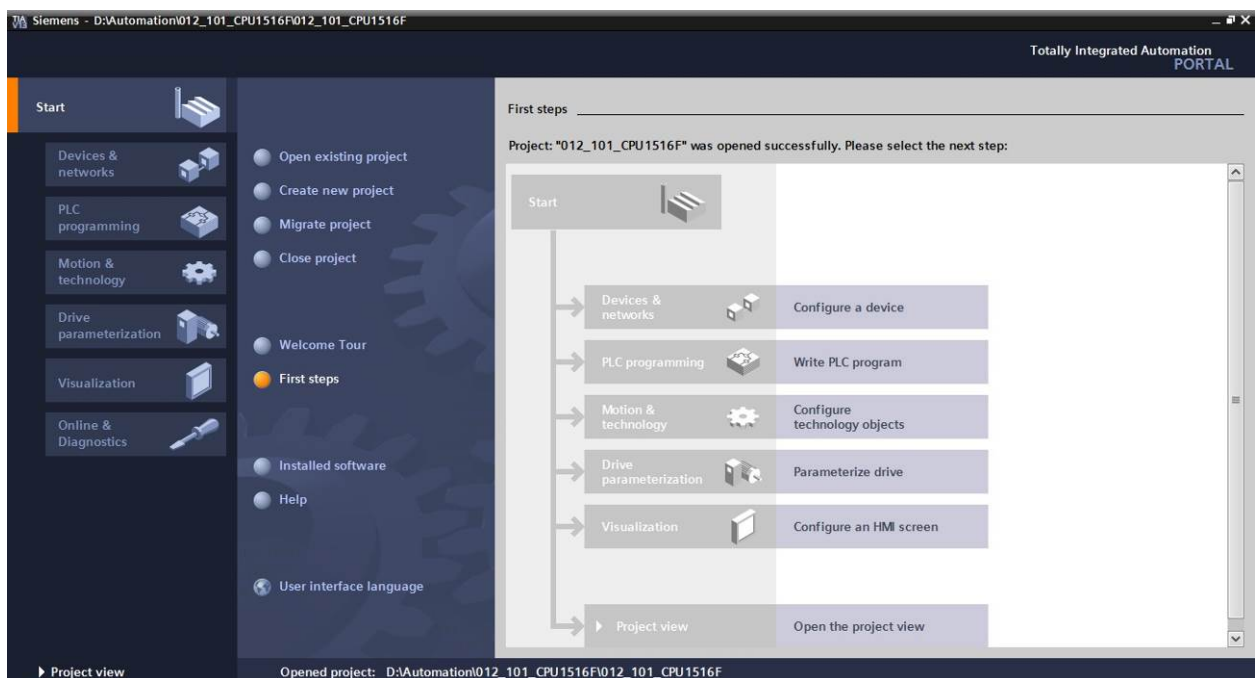


Figure 2: Portal view

The project view, as shown in Figure 3, is used for hardware configuration, programming, creation of the visualization and many other tasks.

By default, the project view displays the menu bar with the toolbars at the top, the project tree with all components of a project on the left and the so-called task cards with instructions and libraries, for example, on the right.

If an element (for example, the device configuration) is selected in the project tree, it is displayed in the center and can be worked on there.

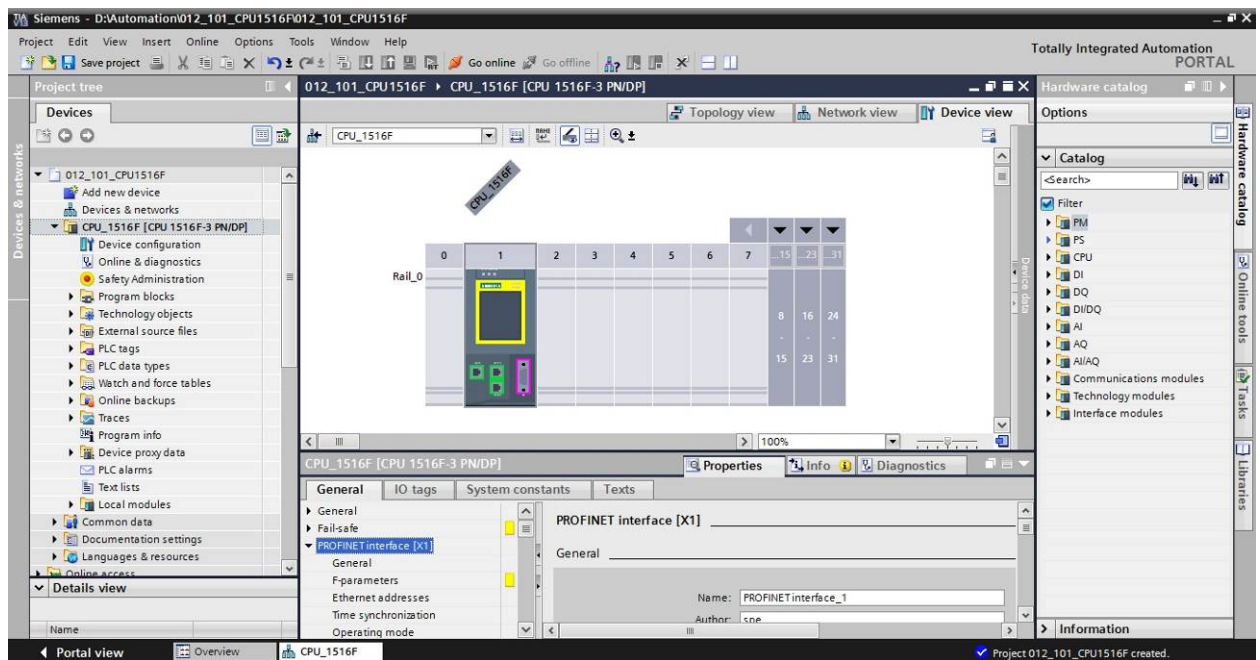
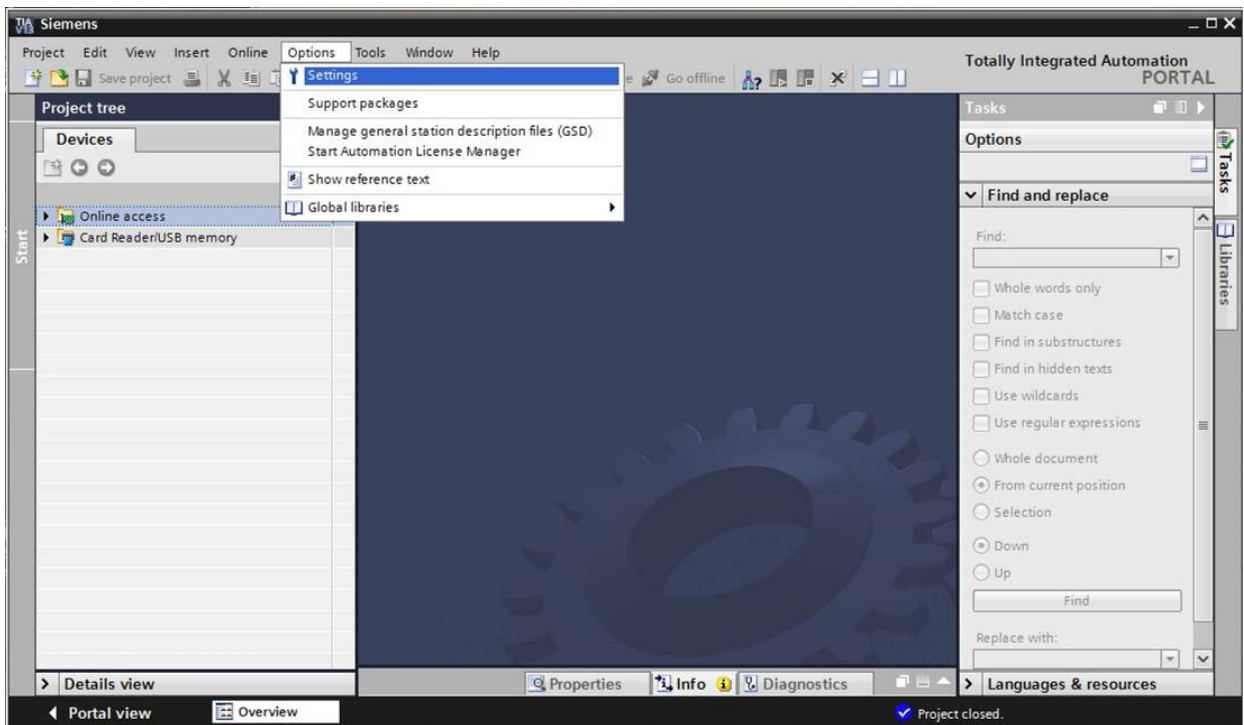


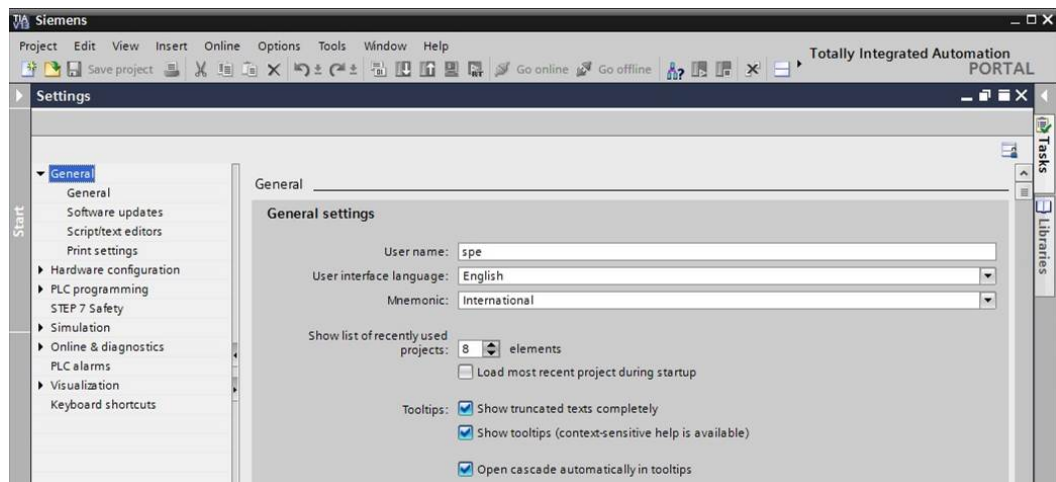
Figure 3: Project view

3.4.6 Basic settings for the TIA Portal

- Users can specify their own default settings for certain settings in the TIA Portal. A few important settings are shown here.
- In the project view, select the →"Options" menu and then → "Settings".

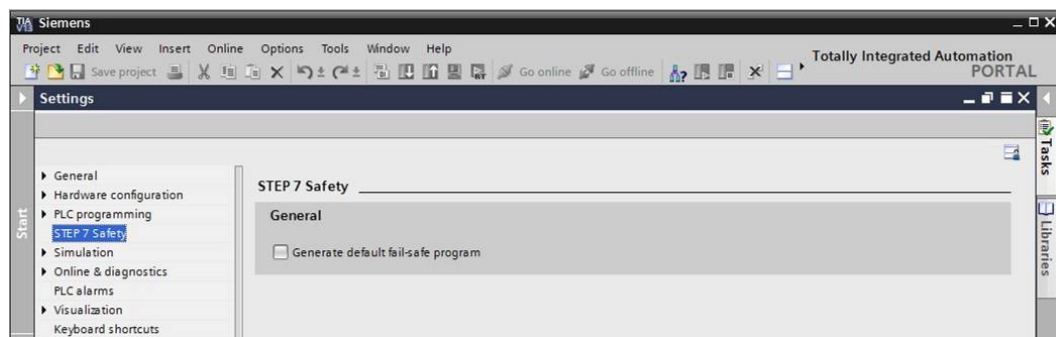


- One basic setting is the selection of the user interface language and the language for the program display. In the curriculums to follow, "English" will be used for both settings.
- Under → "General" in "Settings", select "User interface language → English" and "Mnemonic → International".



Note: These settings can always be changed.

- When Safety CPUs are used (e.g. CPU 1516F-3 PN/DP) without the use of safety engineering, it is recommended that automatic creation of the safety program be deactivated before creating a project.
- In "Settings" under the → "STEP 7 Safety" item, deactivate → "Generate default fail-safe program".




3.4.7 Setting the IP address on the programming device

To program SIMATIC S7-1500 from the PC, the programming device or a laptop, you need a TCP/IP connection or an optional PROFIBUS connection.

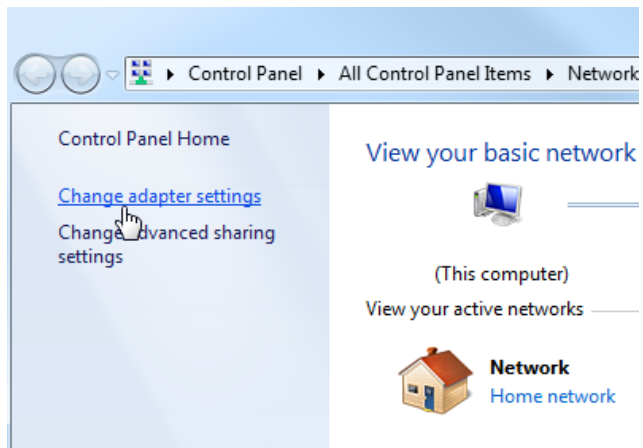
For the PC and SIMATIC S7-1500 to communicate with each other via TCP/IP, it is important that the IP addresses of both devices match.

First, we show you how to set the IP address of a computer with the Windows 7 operating system.

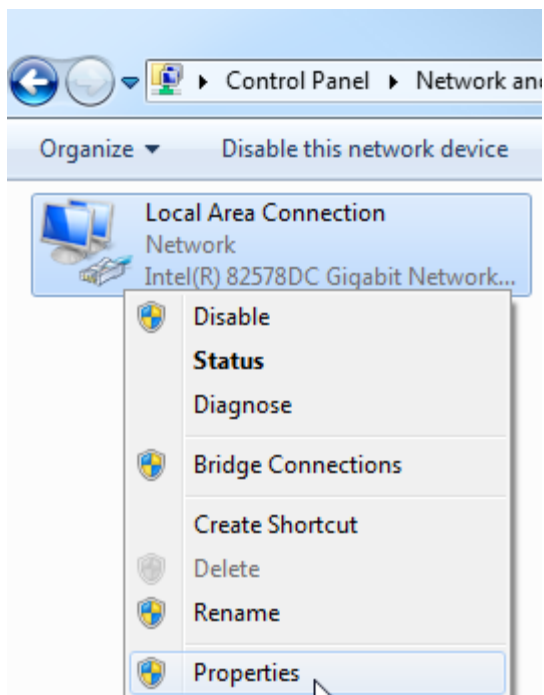
- Locate the network icon in the taskbar at the bottom  and click → "Open Network and Sharing Center".



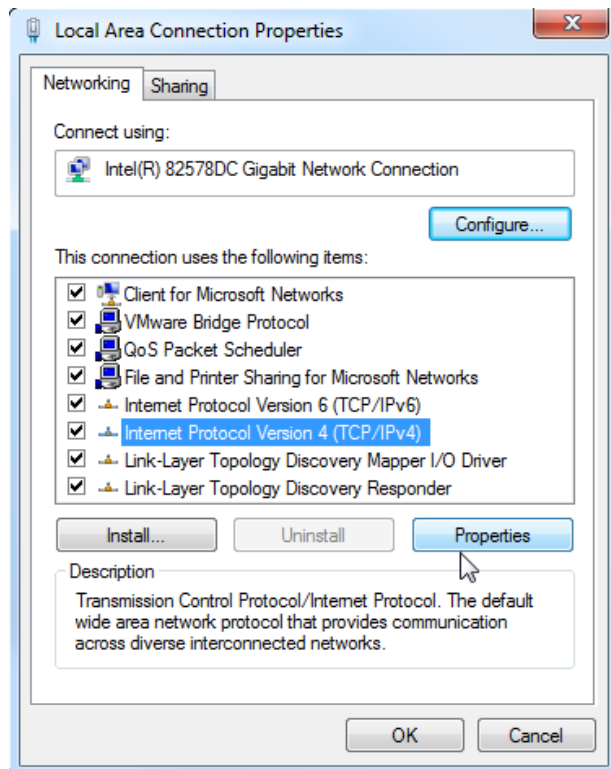
→ In the open Network and Sharing Center window, click → "Change adapter settings".



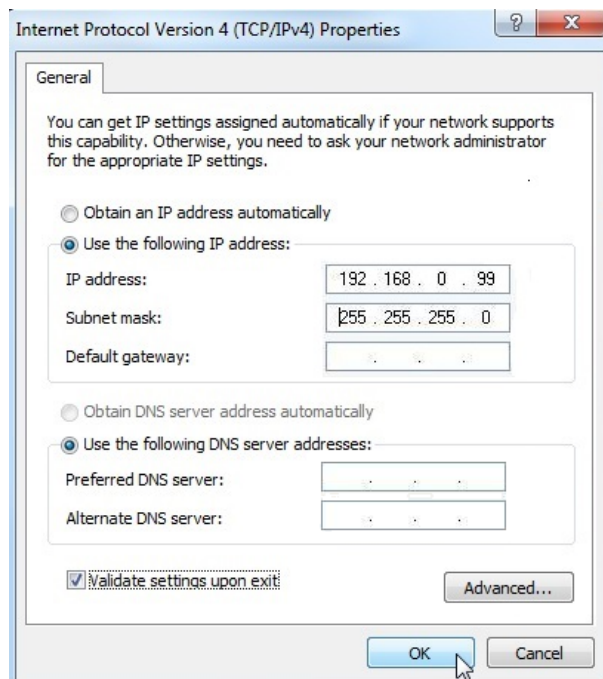
→ Select the desired → "Local Area Connection" that you want to use to connect to the controller and click → "Properties".



→ Next, select → "Properties" for → "Internet Protocol Version 4 (TCP/IP)".



→ You can use the following IP address, for example → IP address: 192.168.0.99 → Subnet mask 255.255.255.0 and accept the settings (→ "OK")



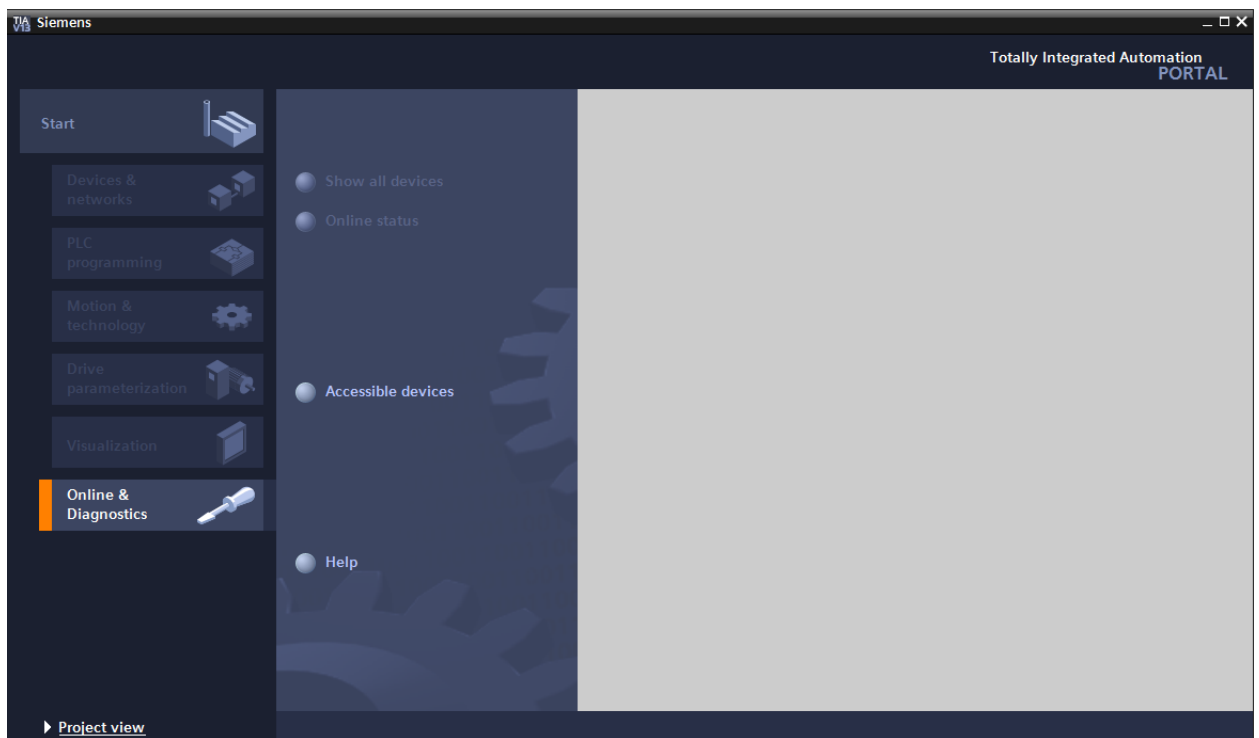
3.4.8 Setting the IP address in the CPU

The IP address of SIMATIC S7-1500 is set as follows.

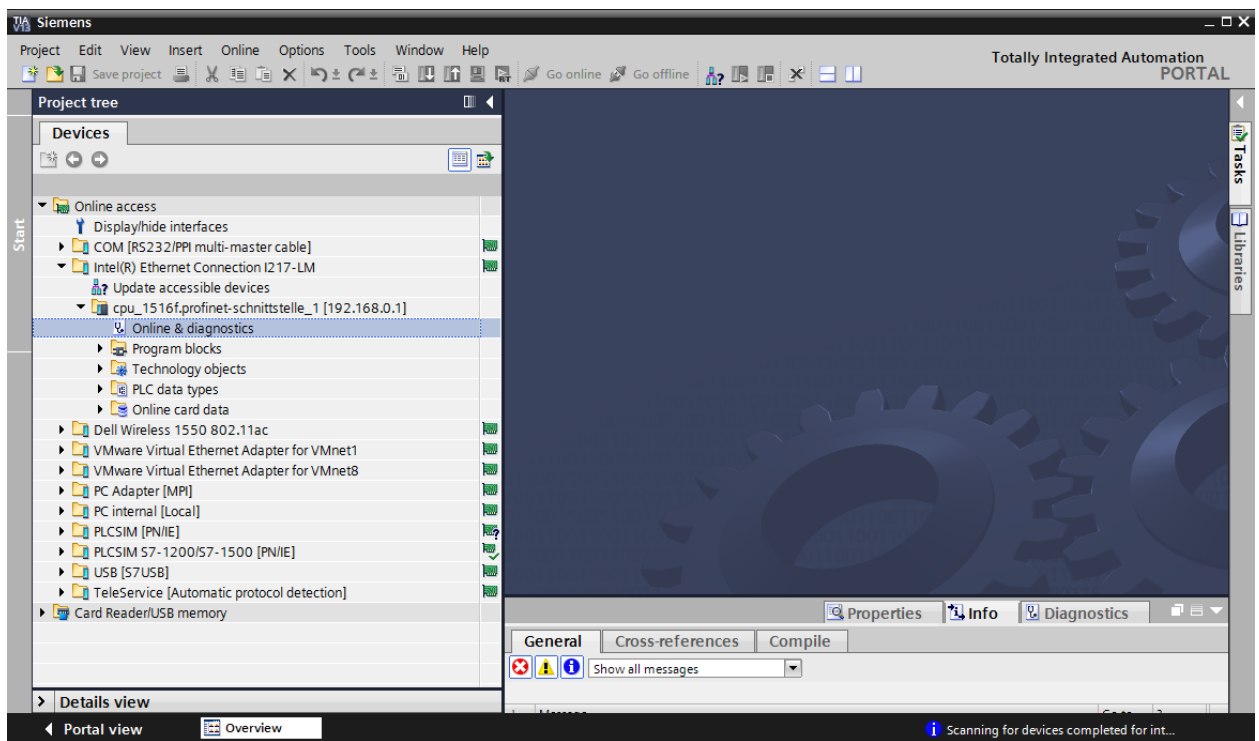
- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V13)



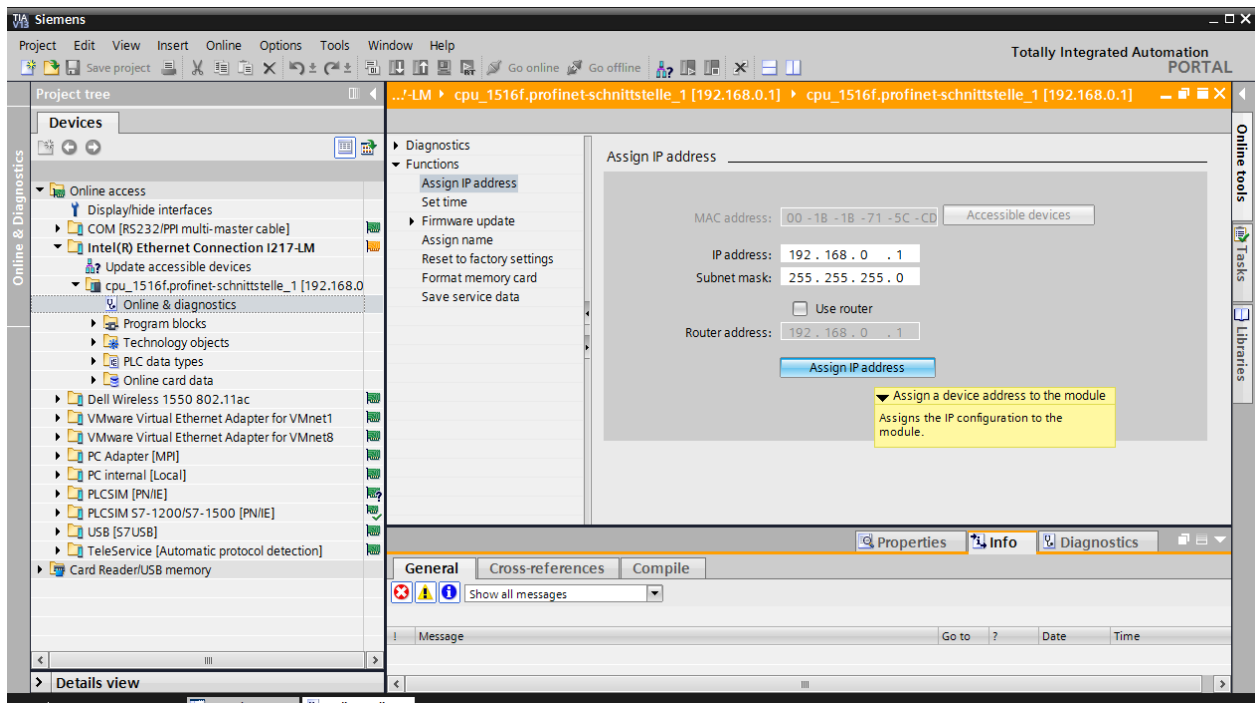
- Select → "Online & Diagnostics" and open the → "project view".



- In the project tree under → "Online access", select the network adapter that was set previously. If you click → "Update accessible devices" here, you will see the IP address (if previously set) or the MAC address (if IP address not yet assigned) of the connected SIMATIC S7-1500. Select → "Online & Diagnostics" here.

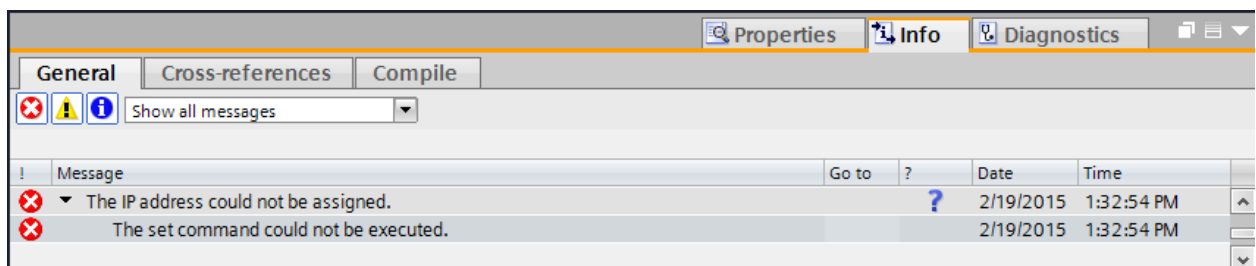


- Under → "Functions", you now find the → "Assign IP address" item. Enter the following IP address here (example): → IP address: 192.168.0.1 → Subnet mask 255.255.255.0. Next, click → "Assign IP address" and this new address will be assigned to your SIMATIC S7-1500.



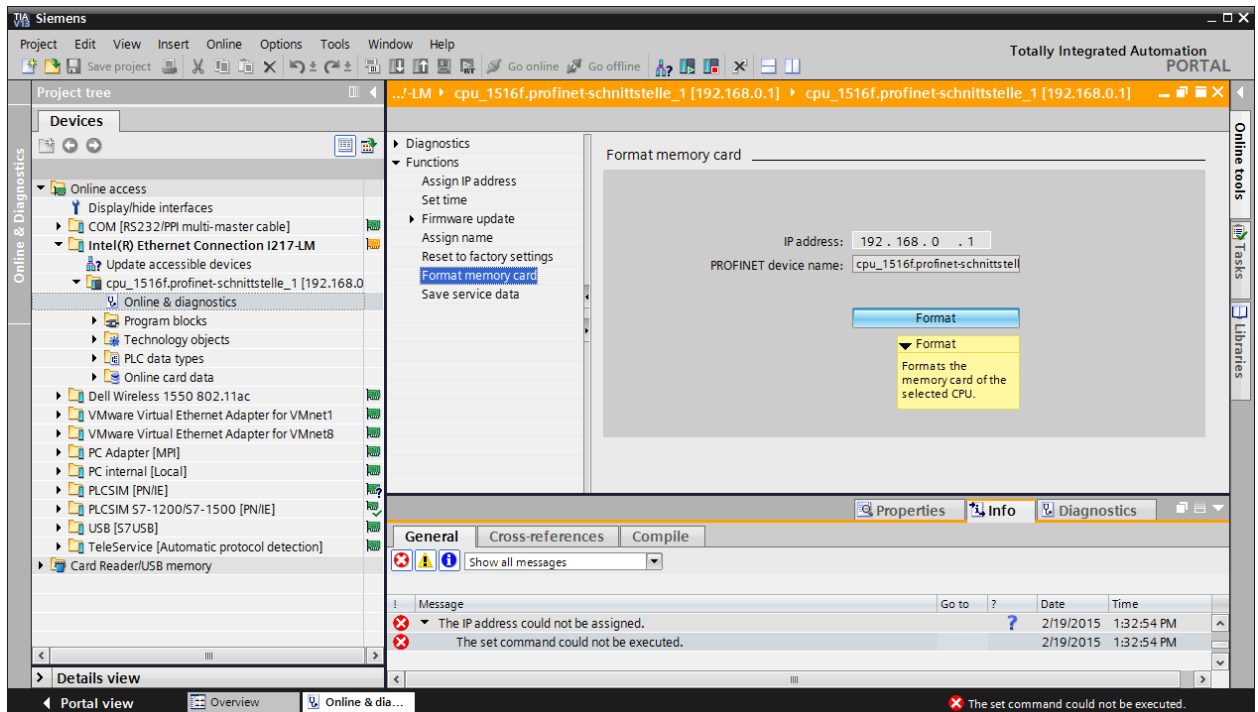
Note: The IP address of the SIMATIC S7-1500 can also be set via the display on the CPU, provided this is enabled in the hardware configuration.

- If the IP address was not successfully assigned, you will receive a message in the → "Info" window under → "General".

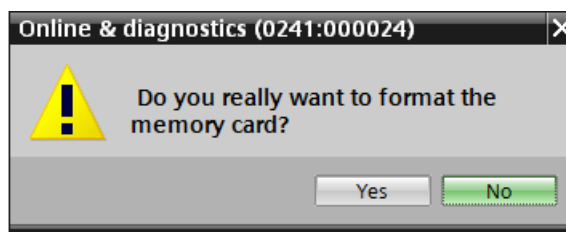


3.4.9 Formatting the memory card in the CPU

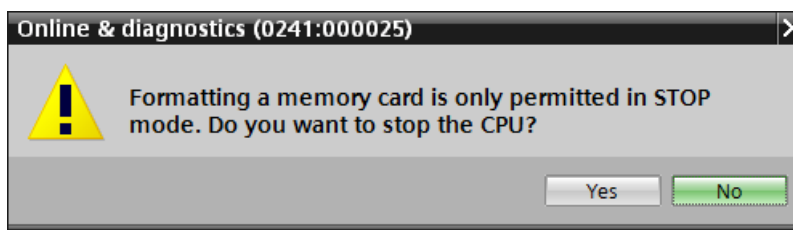
- If the IP address could not be assigned, the program data on the CPU must be deleted. This is accomplished in 2 steps: → "Format memory card" and → "Reset to factory settings".
- First, select the → "Format memory card" function and press the → "Format" button.



- Confirm the prompt asking if you really want to format the memory card with → "Yes".

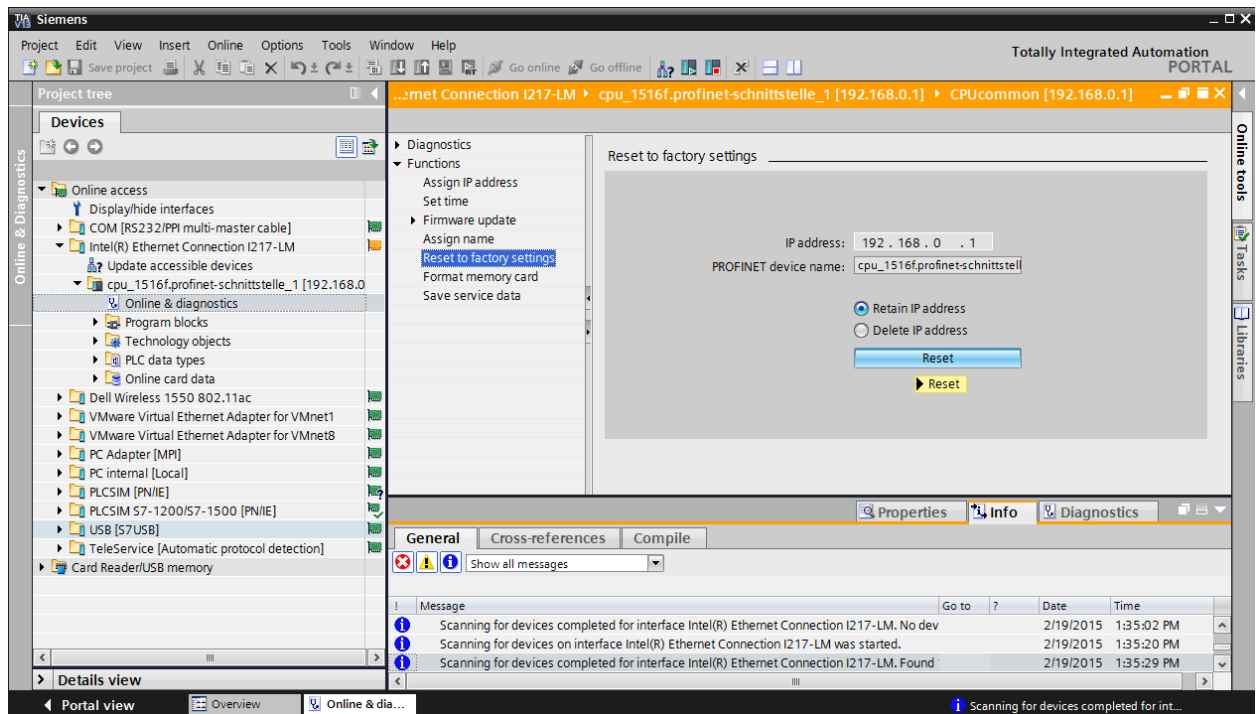


- If necessary, stop the CPU. (→ "Yes")

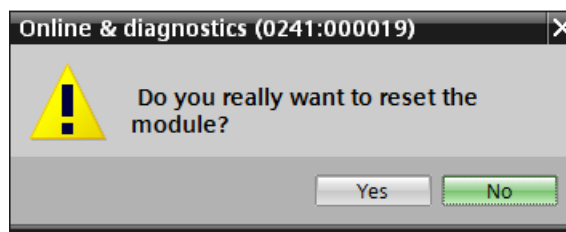


3.4.10 Resetting the CPU to factory settings

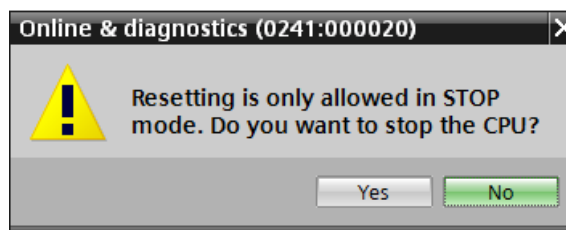
- Before you can reset the CPU, you must wait until the formatting in the CPU has finished. Then you must select → "Update accessible devices" and → "Online & diagnostics" of your CPU again. To reset the controller, select the → "Reset to factory settings" function and click → "Reset".



- Confirm the prompt asking if you really want to reset the module with → "Yes".



- If necessary, stop the CPU. (→ "Yes")



4 Task

Create a project und configure the following modules of your hardware, which correspond to one part of the Trainer Package **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**.

- 1X SIMATIC PM 1507 24 V/8 A STABILIZED POWER SUPPLY INPUT: 120/230 V AC OUTPUT: 24 V DC / 8 A (order number: 6EP1333-4BA00)
- SIMATIC S7-1500, CPU 1516F-3 PN/DP, WORK MEMORY 1.5 MB PROGRAM, 5 MB DATA, 1. INTERFACE, PROFINET IRT WITH 2 PORT SWITCH, 2. INTERFACE, ETHERNET, 3. INTERFACE, PROFIBUS, 10 NS BITPERFORMANCE, SIMATIC MEMORY CARD REQUIRED (order number: 6ES7 516-3FN01-0AB0)
- 1X SIMATIC S7-1500, DIGITAL INPUT MODULE DI 32 X DC24V, 32 CHANNELS IN GROUPS OF 16 (order number: 6ES7 521-1BL00-0AB0)
- 1X SIMATIC S7-1500, DIGITAL OUTPUT MODULE DQ 32 X DC24V / 0.5A; 32 CHANNELS (order number: 6ES7 522-1BL01-0AB0)

5 Planning

Because this is a new system, a new project must be created.

The hardware for this project is already specified with the SIMATIC S7-1516F PN/DP Trainer Package. Therefore, a selection does not have to be made. Instead, the listed modules of the Trainer Package only have to be inserted in the project. To ensure that the correct modules are inserted, the order numbers from the task should be re-checked directly on the installed device.

Usually the CPU is added first and then the signal modules. The power supply can be added last. See Table 1.

The Ethernet interface must be set for the configuration of the CPU. Other settings for fail-safe operation and password protection are also made. The address area is set for the digital input and output modules.

Module	Order number	Slot	Address area
CPU 1516F-3 PN/DP	6ES7 516-3FN01-0AB0	1	
DI 32x24VDC HF	6ES7 521-1BL00-0AB0	2	0...3
DQ 32 X DC24V / 0.5A HF	6ES7 522-1BL01-0AB0	3	0...3
PM 190W 120/230VAC	6EP1333-4BA00	0	

Table 1: Overview of the planned configuration

As the final step, the hardware configuration must be compiled and downloaded. Any errors present can be detected during compilation and incorrect modules can be detected when the controller is started (*only possible when hardware is present and structured identically*).

The tested project must be saved.

6 Structured step-by-step instructions

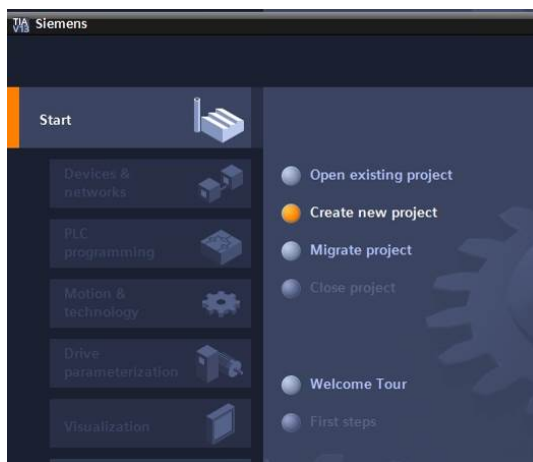
You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it is sufficient to focus on the numbered steps. Otherwise, simply follow the steps of the instructions illustrated below.

6.1 Create a new project

- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V13)



- In the portal view under the "Start" menu, select the command → "Create new project".

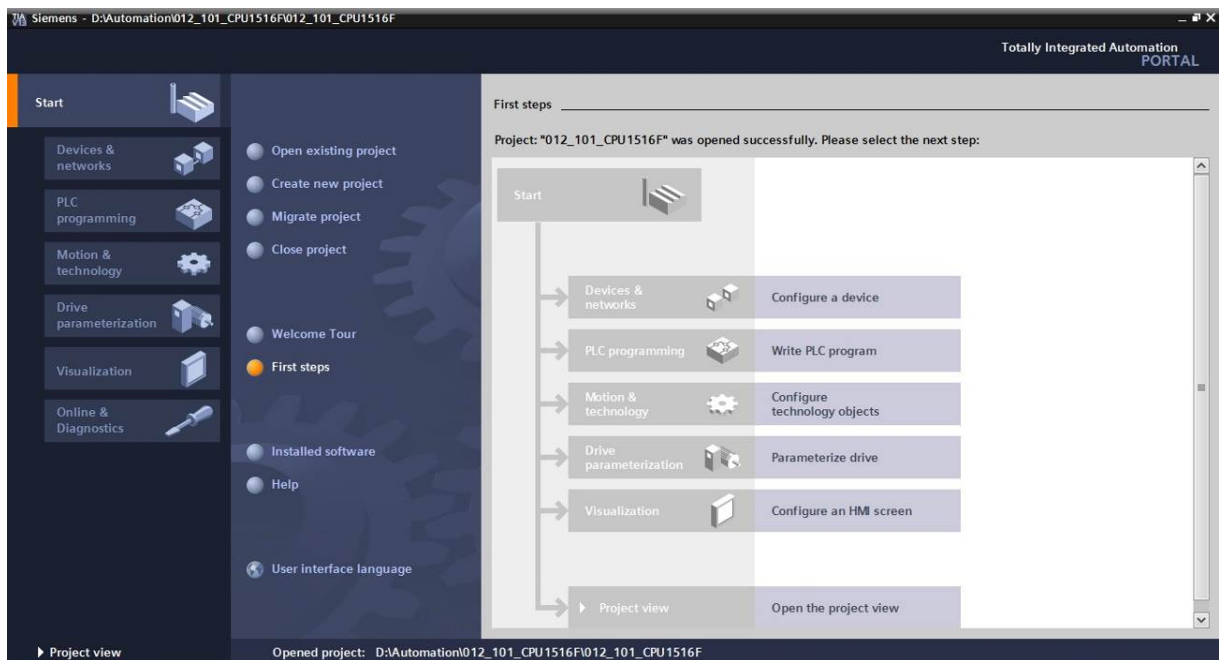


- Modify Project name, Path, Author and Comment as appropriate and click → "Create".

- The project will be created and opened and the menu "Start", "First steps" will open automatically.

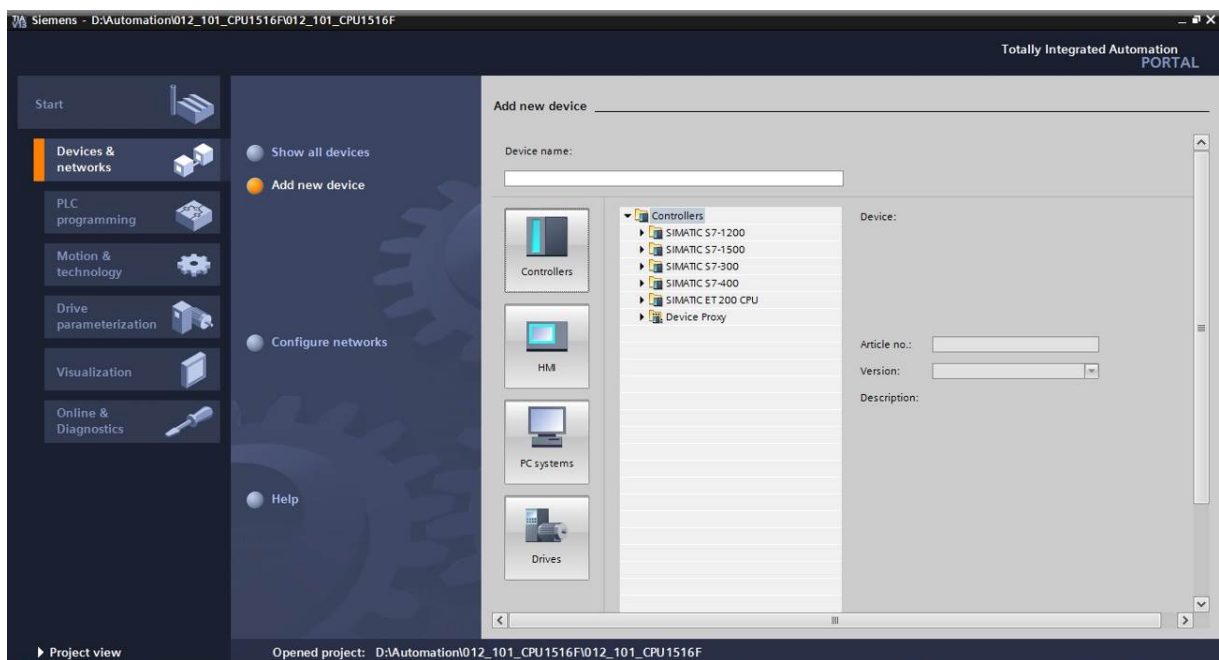
6.2 Insert the CPU 1516F-3 PN/DP

→ In the → "Start" portal, select → "First steps" → "Devices & Networks" → "Configure a device".



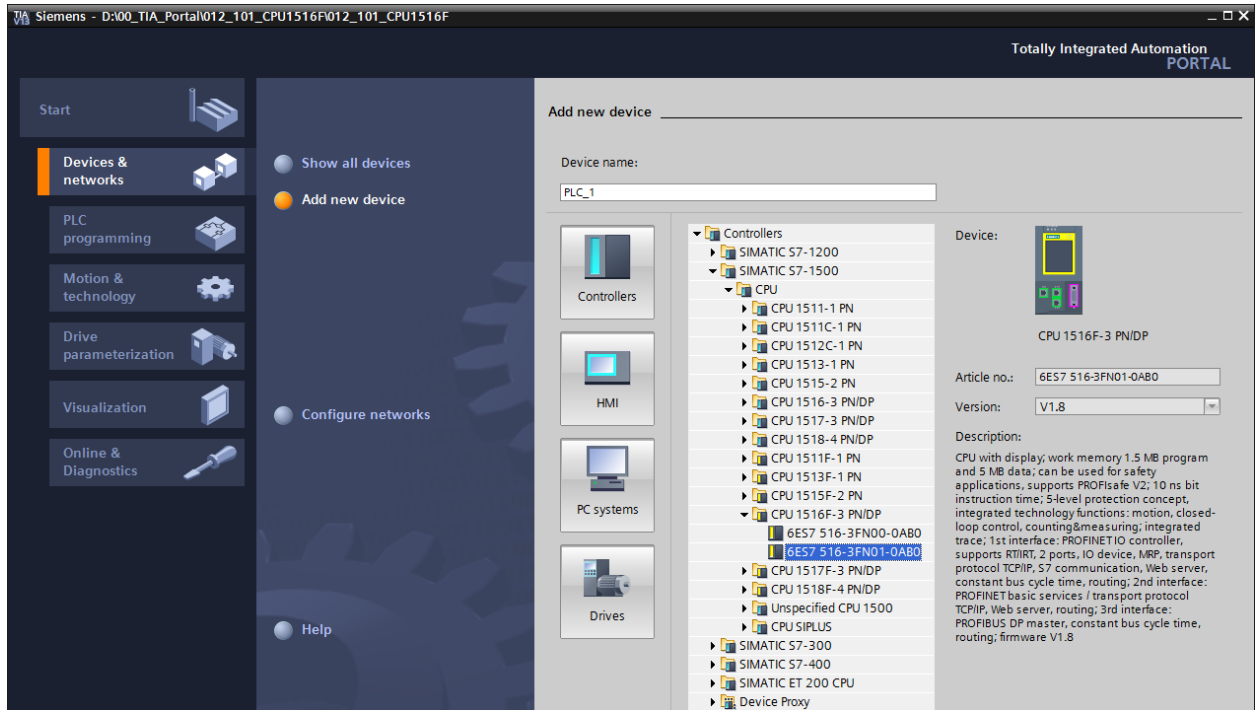
→ The "Show all devices" menu opens in the "Devices & Networks" portal.

→ Switch to the "Add new device" menu.



→ The specified model of the CPU will now be added as a new device.

(Controllers → SIMATIC S7-1500 → CPU → CPU 1516F-3 PN/DP → 6ES7 516-3FN01-0AB0 → V1.8)



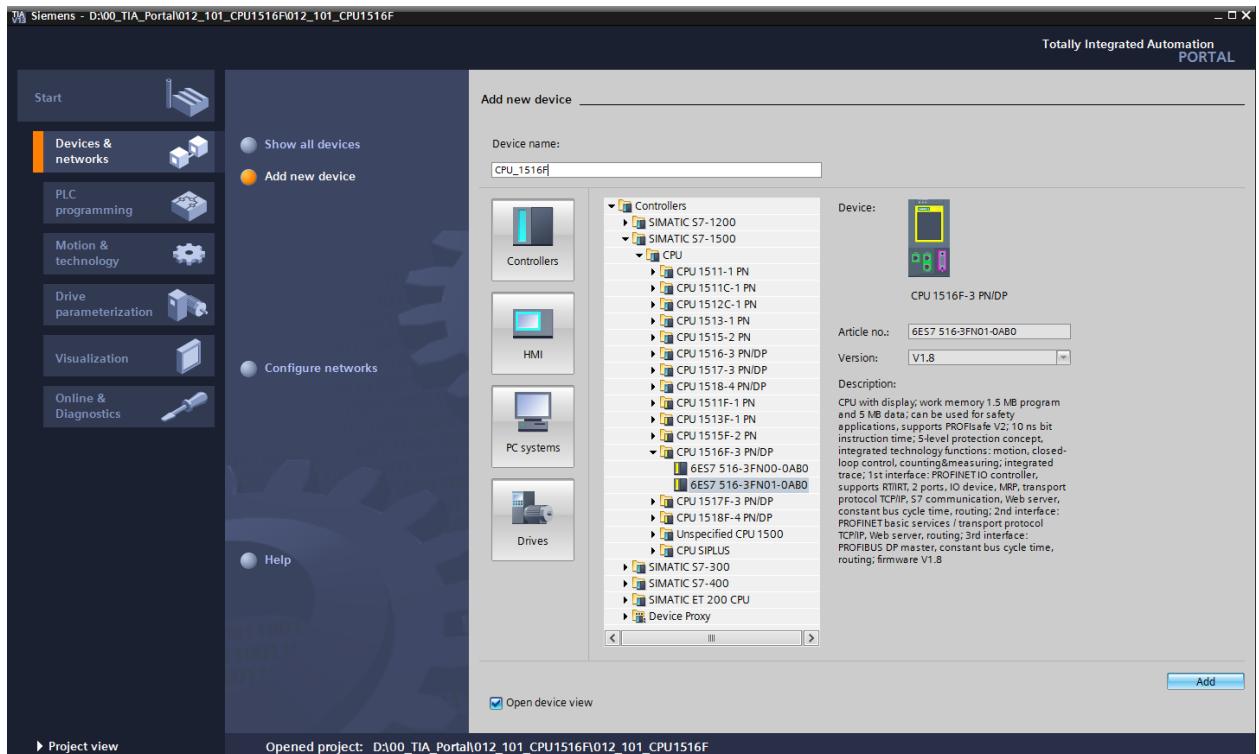
→ Assign a device name (Device name → "CPU_1516F").



→ Select "Open device view".



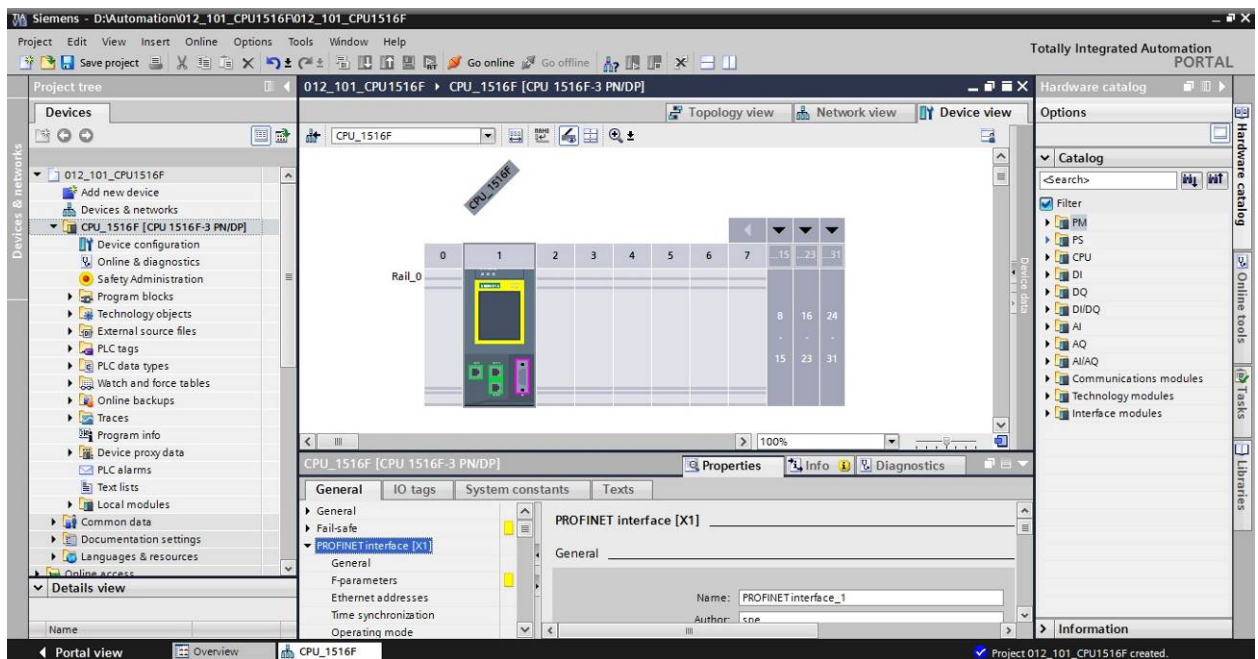
→ Click "Add".



Note: The desired CPU may have multiple versions that differ in functionality (work memory, integrated memory, technology functions, etc.). In this case, you should ensure that the selected CPU meets the requirements placed on it.

Note: Different firmware versions are often offered for the hardware. In this case, it is recommended to use the latest firmware (selected by default).

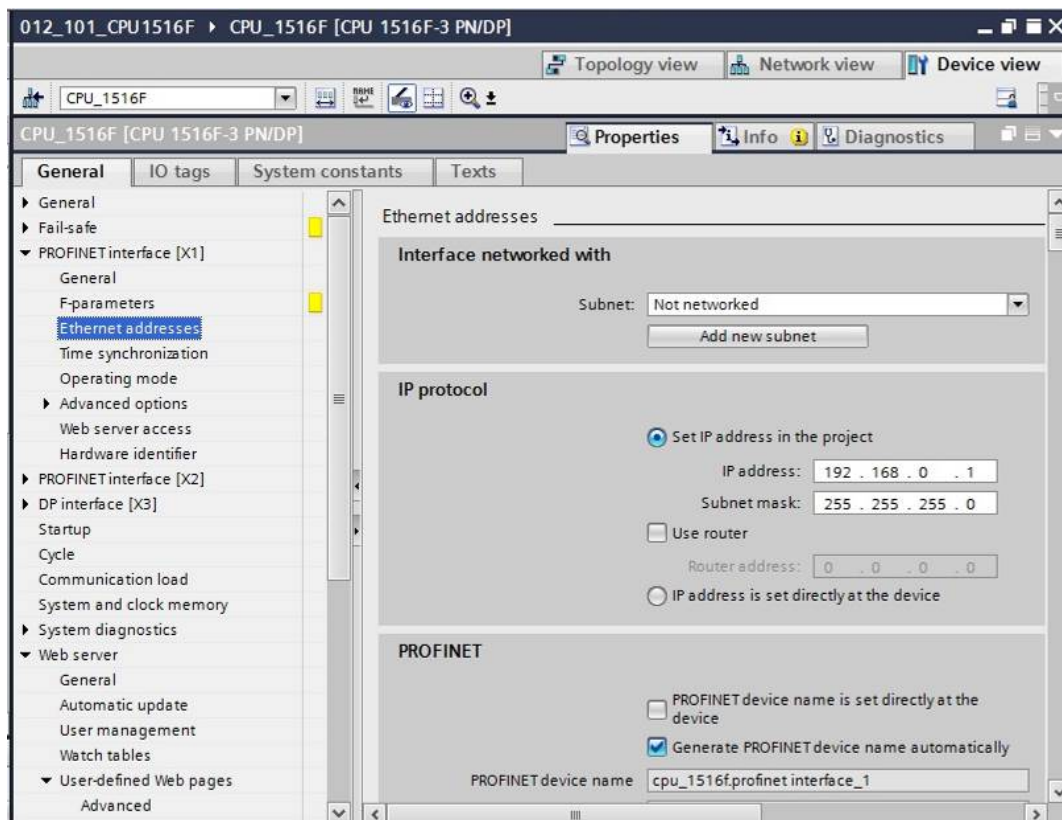
→ The TIA Portal now changes automatically to the project view and displays the selected CPU in the device configuration for slot 1 of a rail.



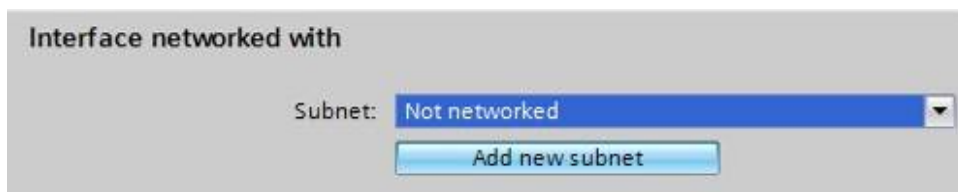
Note: You can now configure the CPU there according to your specifications. Possible settings include the PROFINET and PROFIBUS DP interfaces, startup characteristics, cycle, communication load and many others.

6.3 Configure the Ethernet interface of the CPU 1516F-3 PN/DP

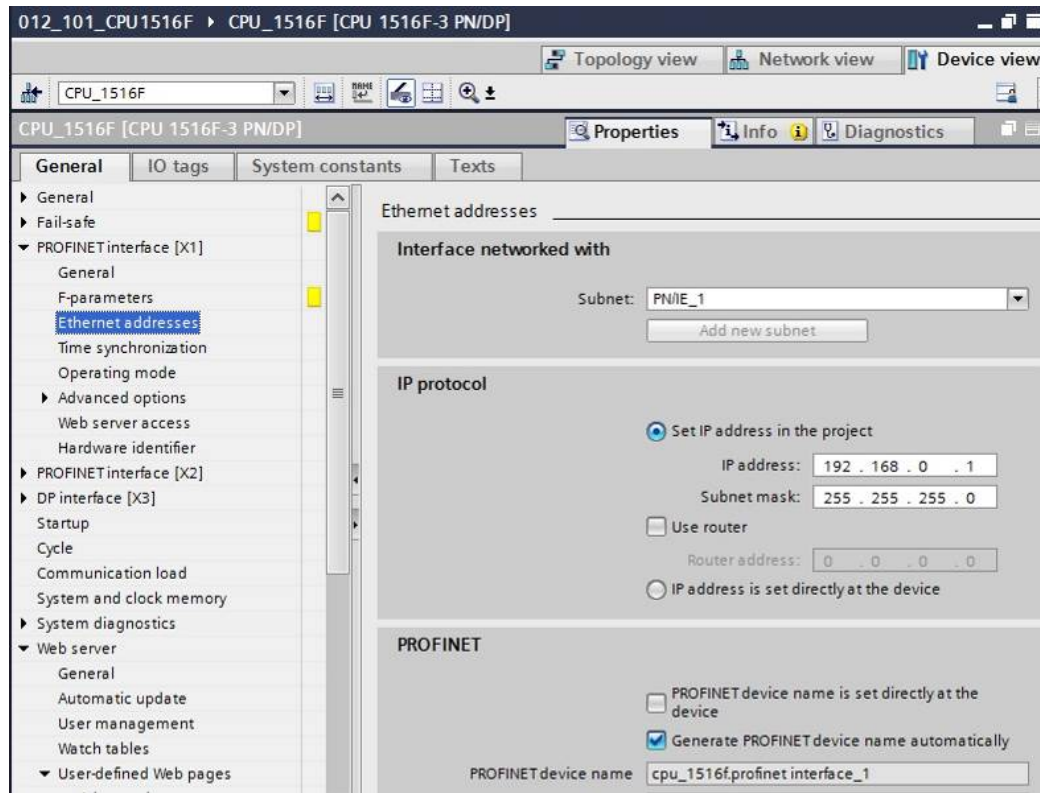
- Select the CPU with a double-click
- Under → "Properties", open the → "PROFINET-interface [X1]" menu and select the → "Ethernet addresses" entry there.



- Under "Interface connected with", only the "Not connected" entry is available.
- Add an Ethernet subnet with the → "Add new subnet" button.



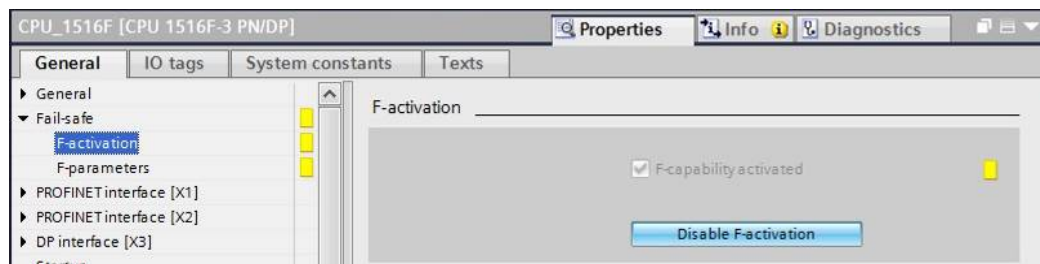
→ Keep the pre-assigned "IP address" and "Subnet mask".



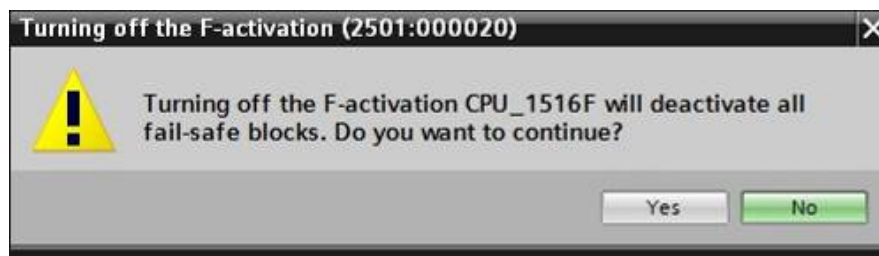
3

6.4 Configure the fail-safe operation of the CPU 1516F-3 PN/DP

→ In the → "Fail-safe" menu, select → "F-activation" and select → "Disable F-activation" there.

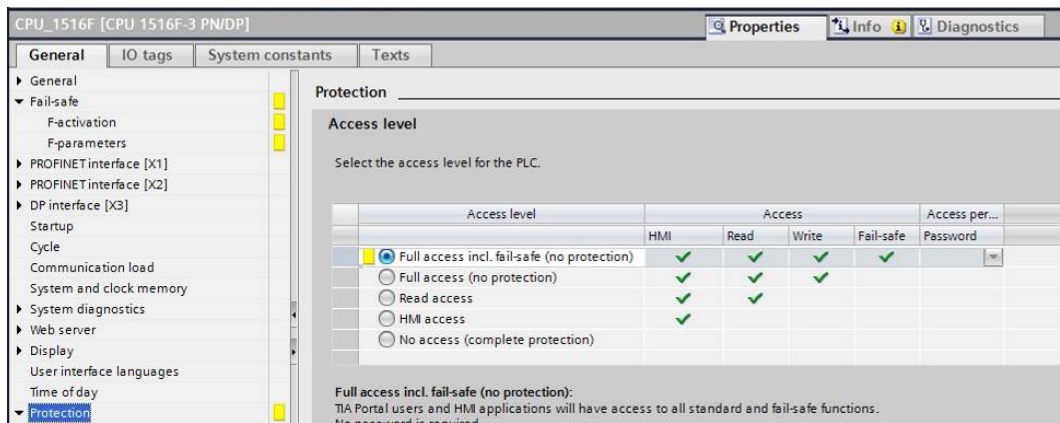


→ Confirm the prompt asking if you want to continue with → "Yes".



6.5 Configure the access level for the CPU 1516F-3 PN/DP

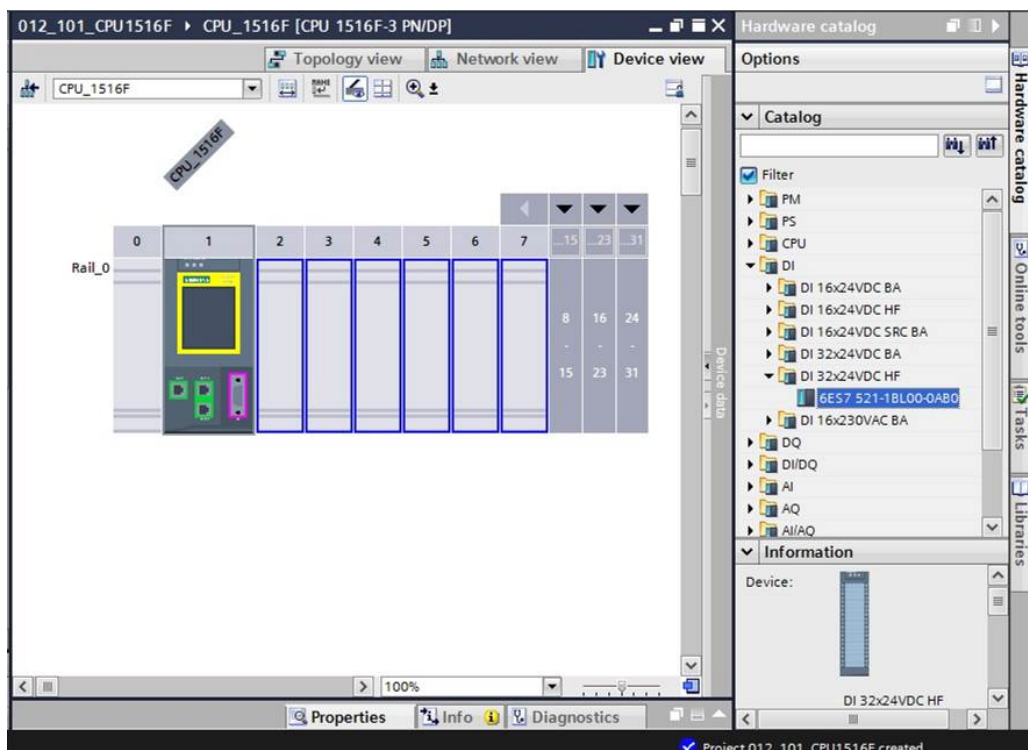
- Switch to the → "Protection" menu and select access level → "Full access incl. fail-safe (no protection)".



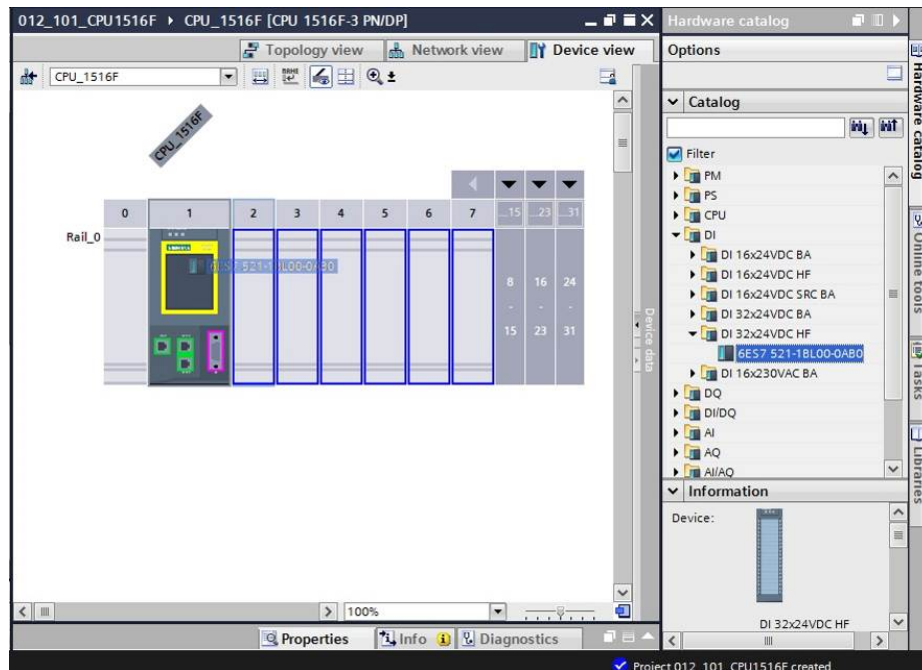
Note: The setting "Full access incl. fail-safe (no protection)" is recommended because a safety program is not created here and thus we also do not have to assign a password.


6.6 Insert the digital input module DI 32x24VDC HF

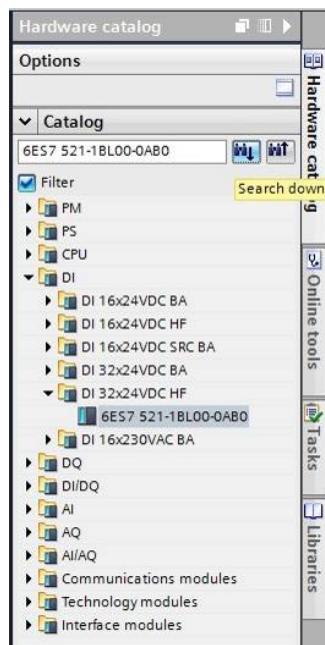
- Find the correct module in the hardware catalog (→ Hardware catalog → DI → DI 32x24VDC HF (order number 6ES7521-1BL00-0AB0))



- Insert the digital input module by using drag-and-drop to move it into slot 2 of the mounting rail.



Note: To select the digital input module, you can simply enter the order number in the Search field and then click the "Search down"  icon. The hardware catalog will open at the correct position.

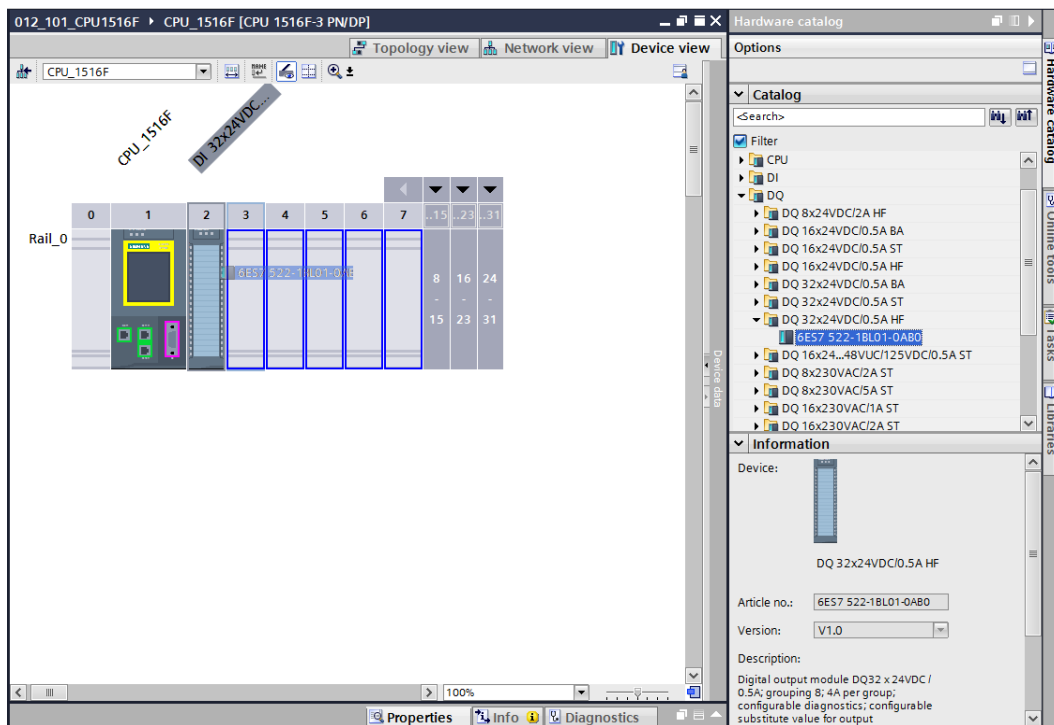


Note: When you double-click a module in the hardware catalog, you insert it at the next available compatible slot.

6.7 Insert the digital output module DQ 32xDC24V / 0.5A HF

- Find the correct module in the hardware catalog (→ Hardware catalog → DQ → DQ 32xDC24V/0.5A HF (order number 6ES7522-1BL01-0AB0))
- Insert the digital output module into slot 3.

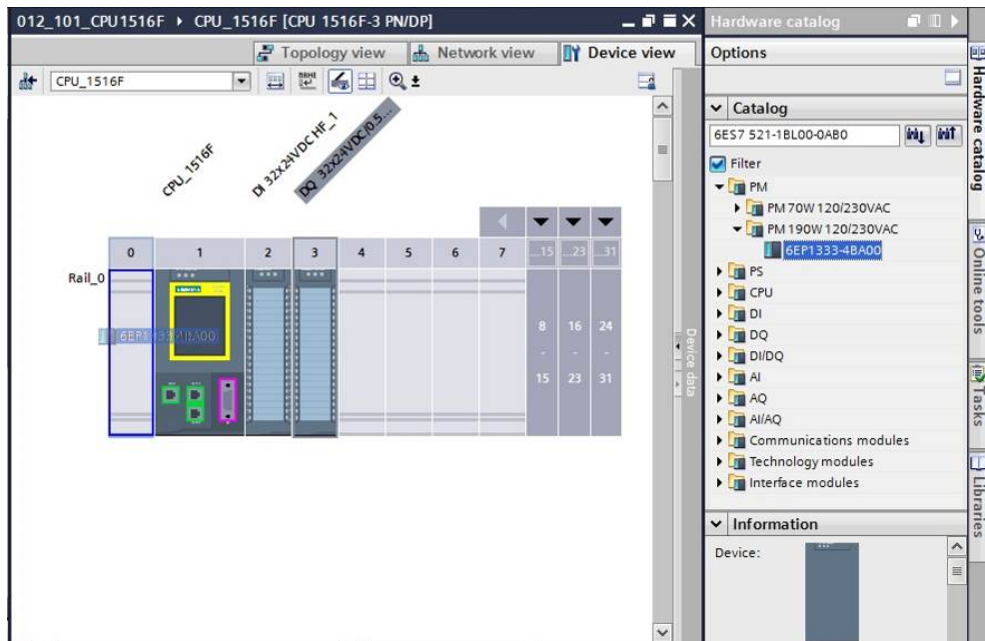
3



Note: If you do not use a slot, you must close the gap before you compile. Otherwise, an error message will occur.

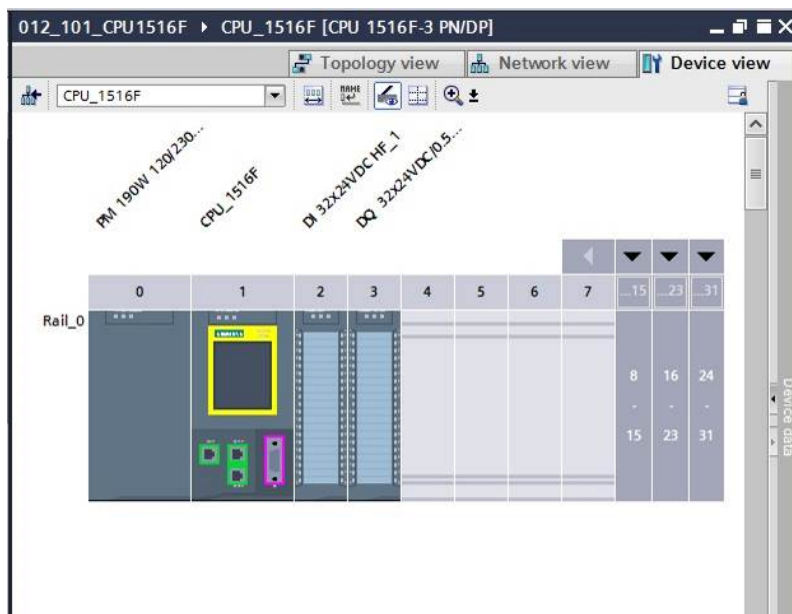
6.8 Insert power module PM 190W 120/230VAC

- Find the correct module in the hardware catalog and insert the power module into slot 0. (→ Hardware Catalog → PM → PM 190W 120/230VAC (order number 6EP1333-4BA00) → Slot 0)



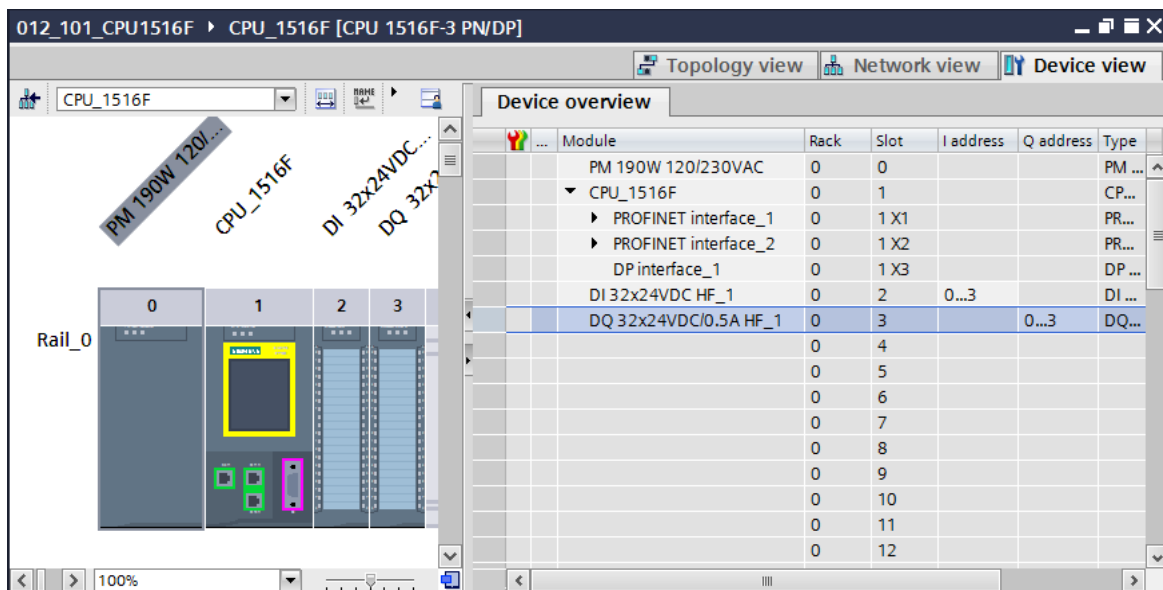
Note: If a module as well as the power module is planned for one slot, it is not possible to place it at another position even in the device configuration.

- Compare your device configuration with the following figure.



6.9 Configure the address areas of the digital input and output modules

- In the "Device overview" area, make certain that the "DI 32x24VDC HF" module has input address area 0...3. (→ Device overview → DI 32x24VDC HF → I address → 0...3)
- Next, check if the "DQ 32xDC24V/0.5A HF" module has output address area 0...3. (→ Device overview → DQ 32xDC24V/0.5A HF → Q address → 0...3)

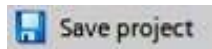


Note: To show and hide the Device overview, you must click the small arrow next to "Device data" on the right side of the hardware configuration.



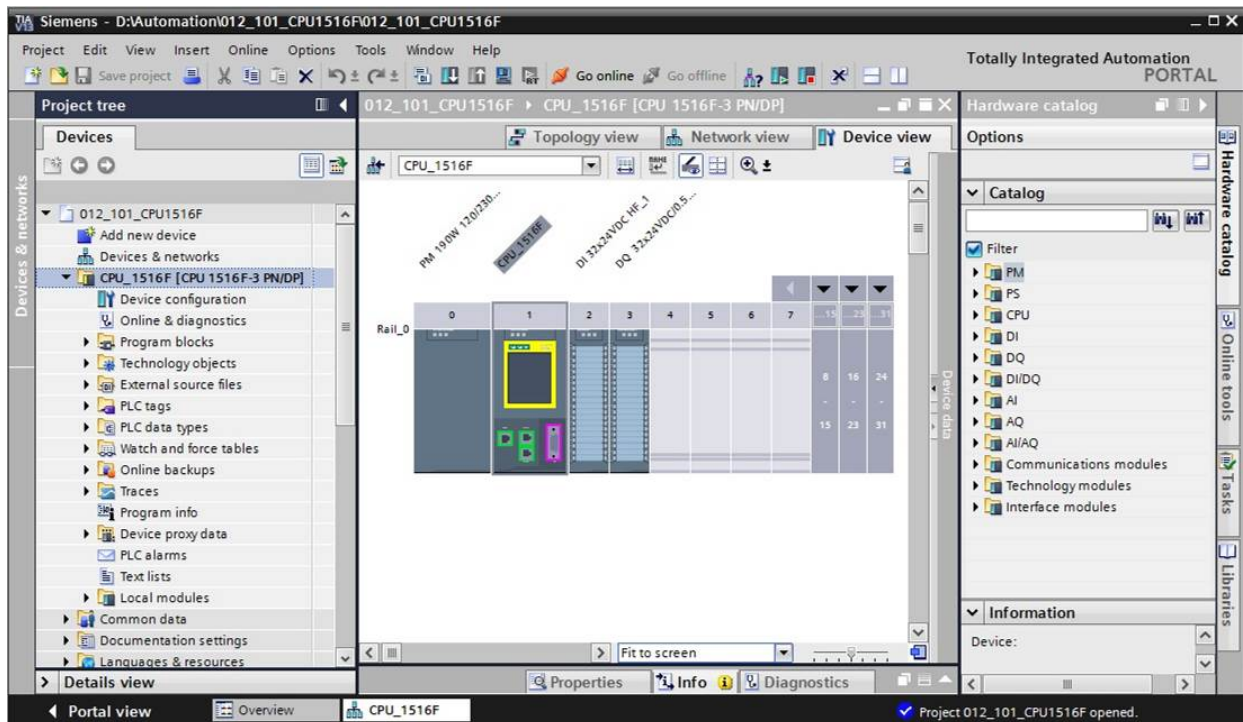
6.10 Save and compile the hardware configuration

→ Before you compile the configuration, you should save your project by clicking the →



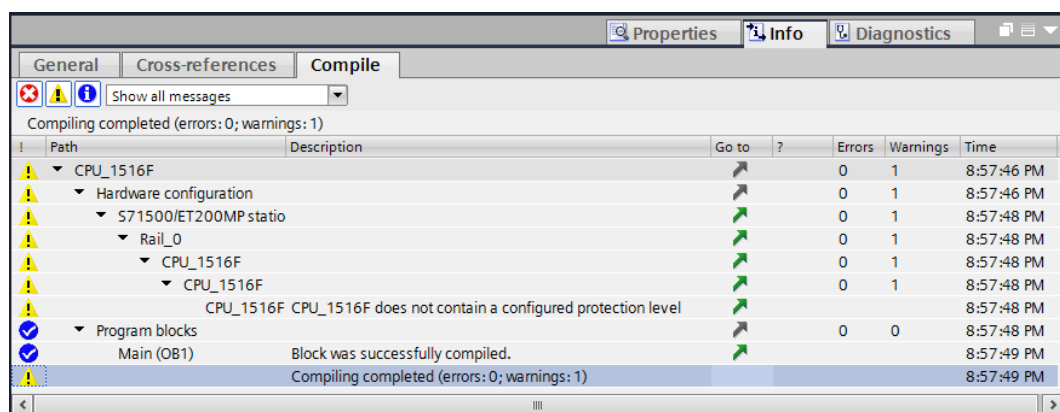
button. To compile your CPU with the device configuration, first select the

→ "CPU_1516F [CPU1516F-3 PN/DP]" folder and click the → "Compile" icon.




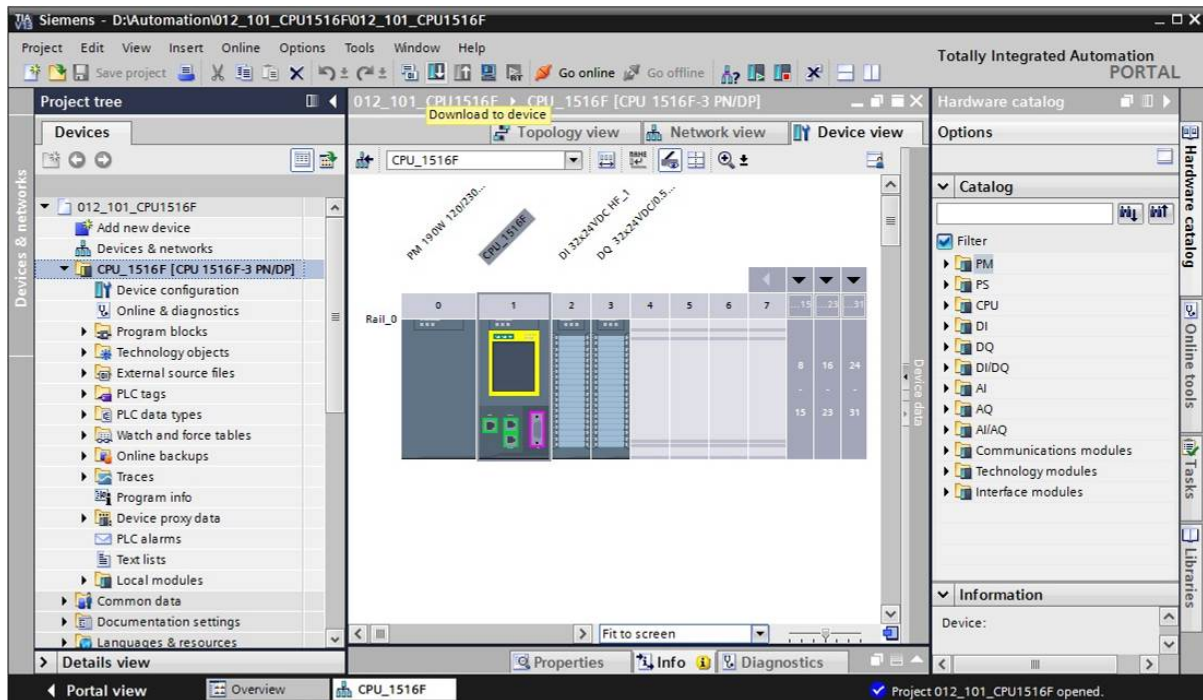
Note: "Save project" should be used repeatedly when working on a project since this does not happen automatically. A prompt to save the project only occurs when the TIA Portal is closed.

→ If the project was compiled without errors, you see the following screen.

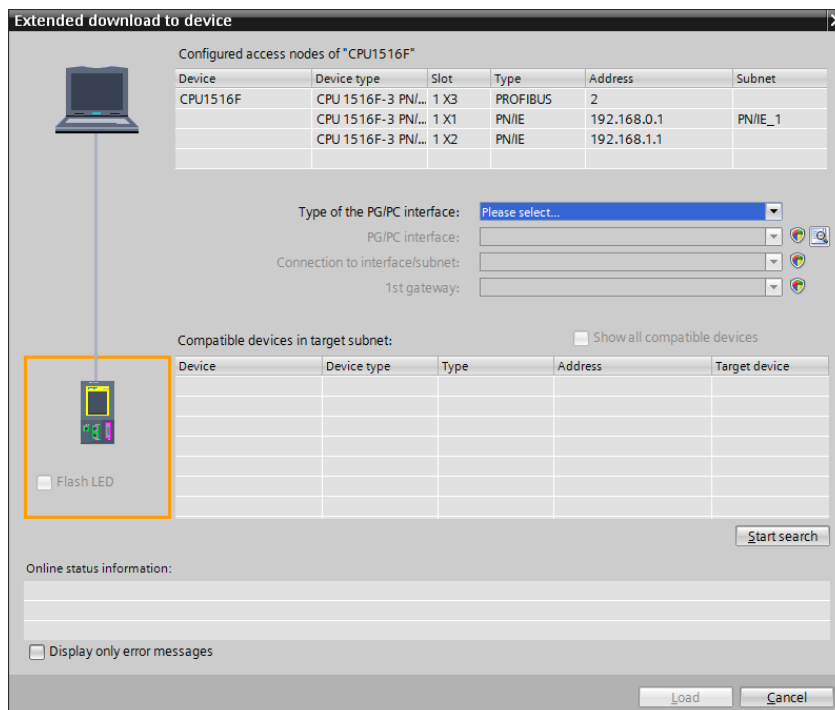


6.11 Download the hardware configuration to the device

→ To download your entire CPU, select the → "CPU_1516F [CPU1516F-3 PN/DP]" folder and click the  → "Download to device" icon.

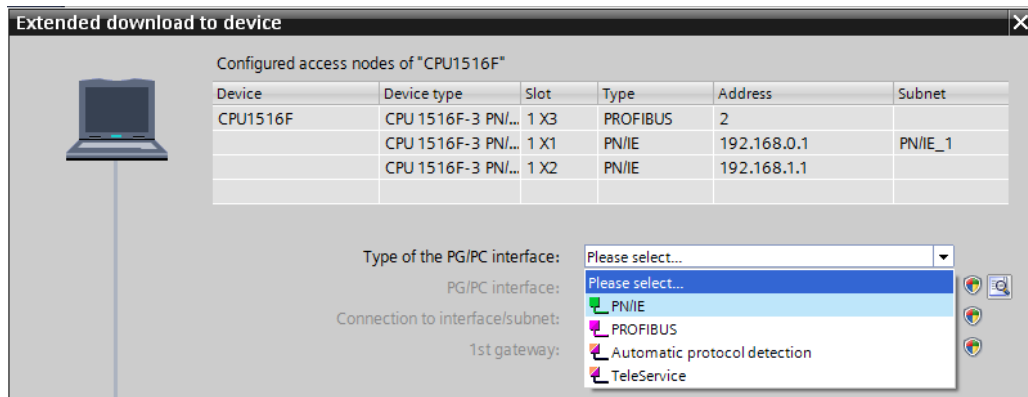


→ The manager for configuring the connection properties (extended download) opens.

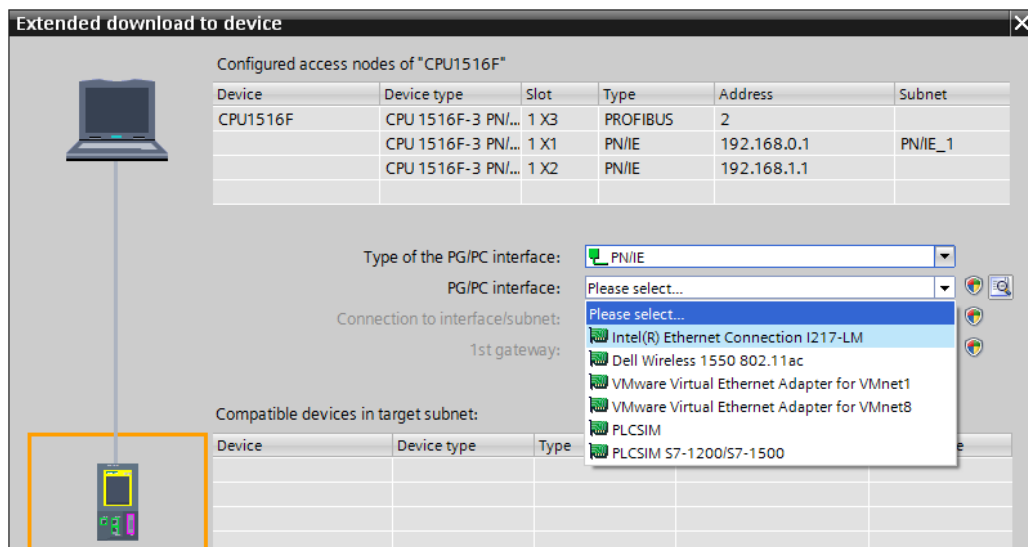


→ First, the interface must be correctly selected. This happens in three steps.

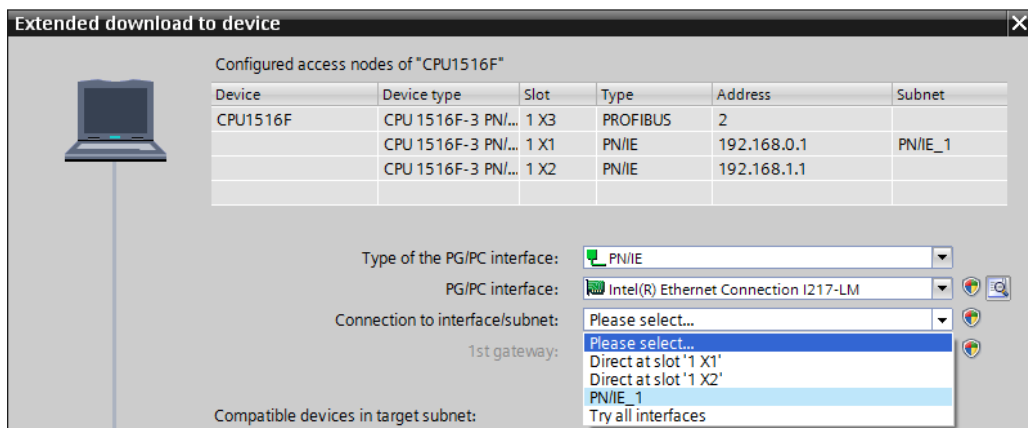
→ Type of the PG/PC interface → PN/IE



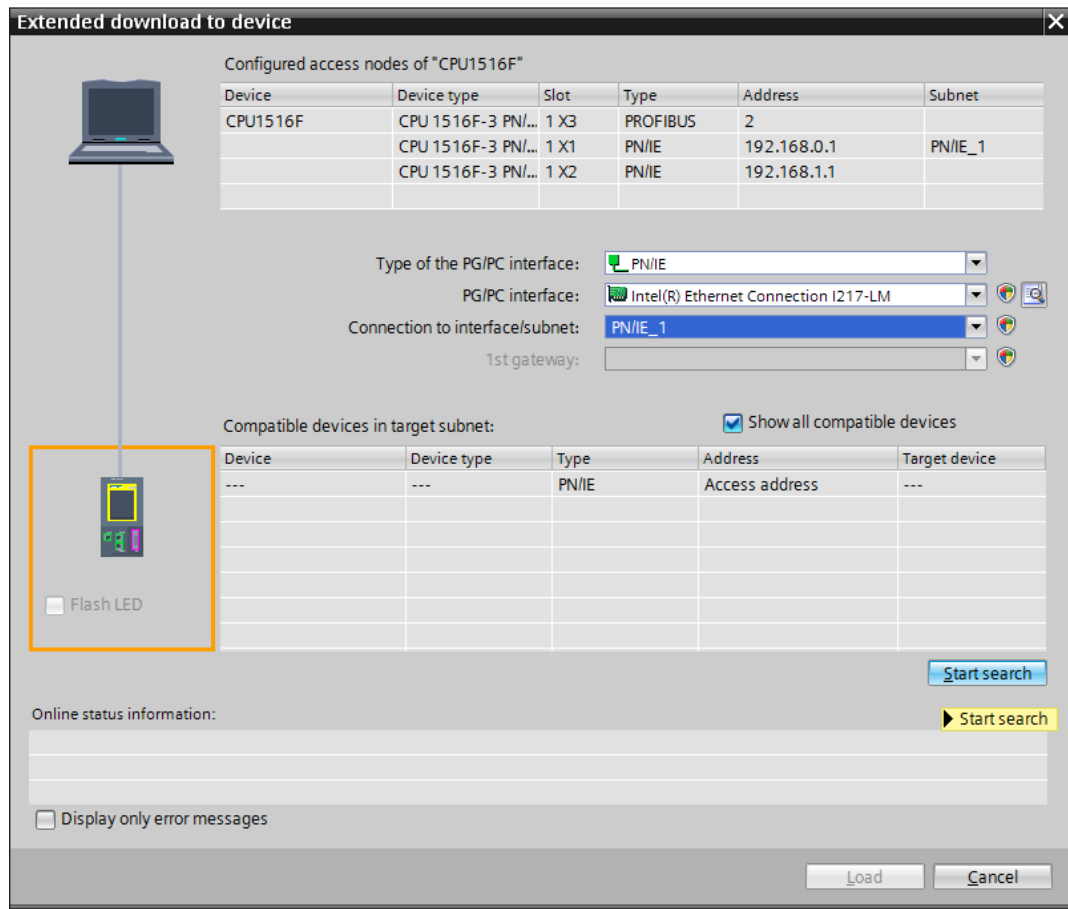
→ PG/PC interface → here: Intel(R) Ethernet Connection I217-LM



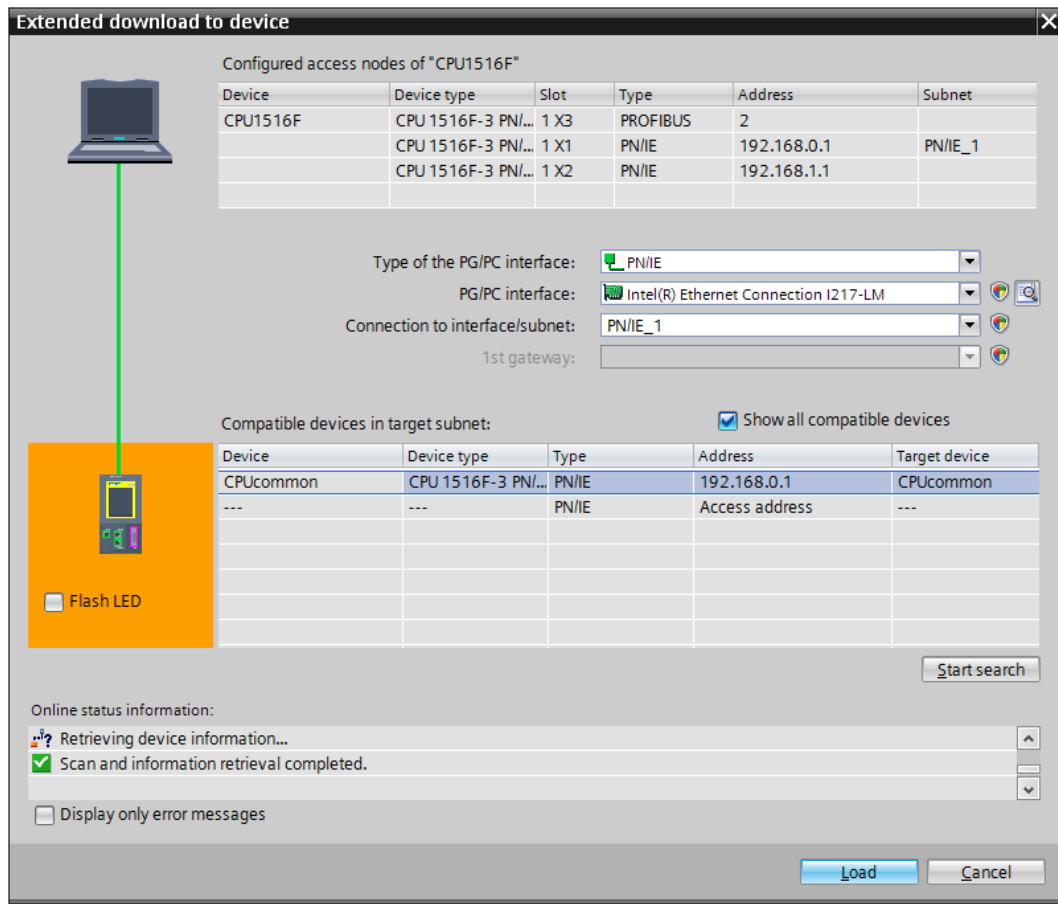
→ Connection to interface/subnet → "PN/IE_1"



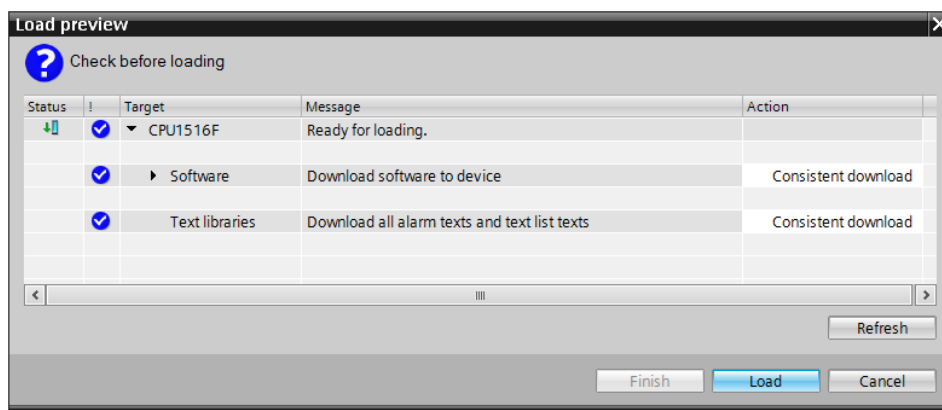
→ The → "Show all compatible devices" check box must be selected. The search for devices in the network is started by clicking the → **Start search** button.



- If your CPU is shown in the "Compatible devices in target subnet" list, it must be selected and the download started. (→ CPU 1516F-3 PN/DP → "Load")

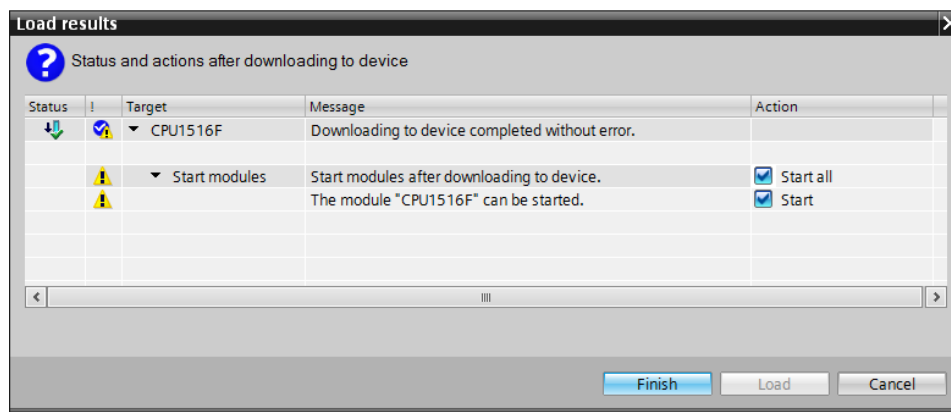


- You first obtain a preview. Confirm the prompt → "Overwrite all" and continue with → "Load".

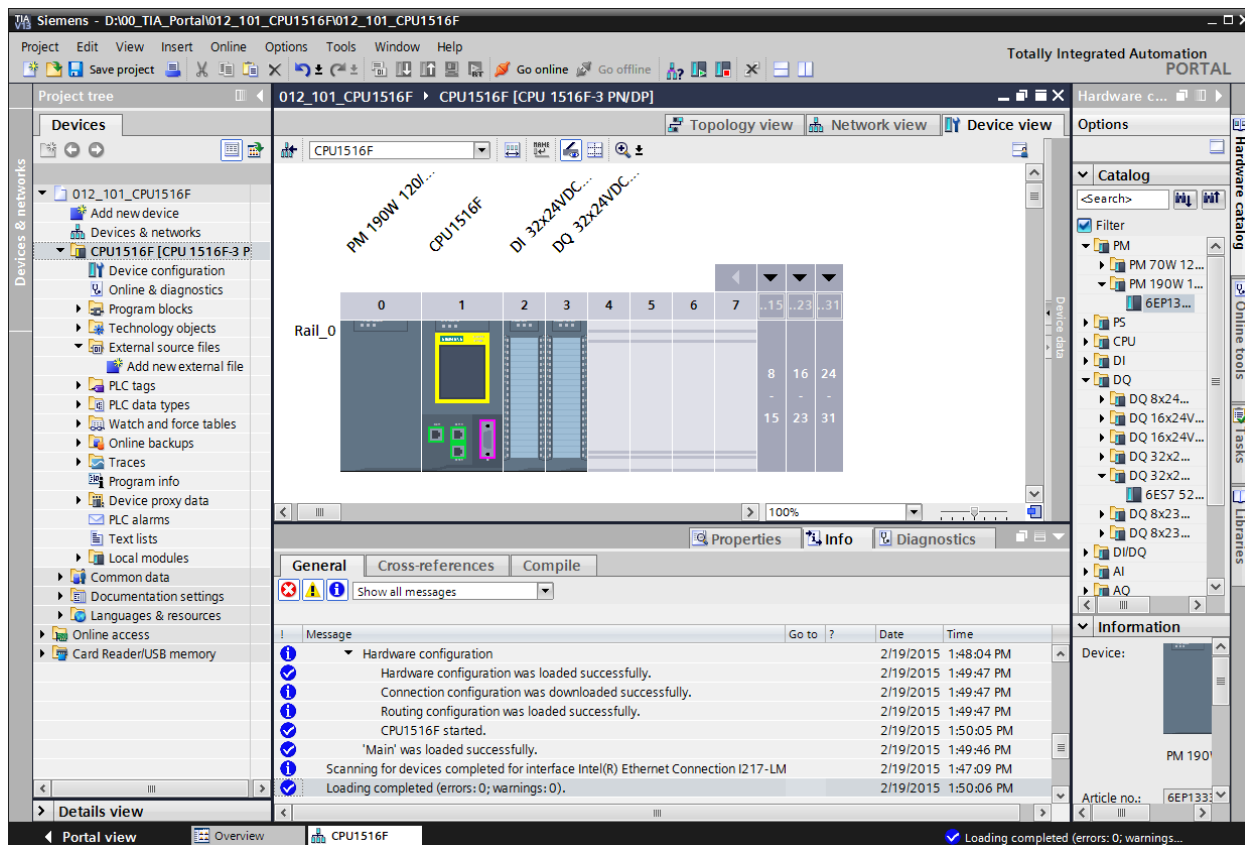


Note: The ✓ symbol should be visible in every line of the "Load preview". You can find additional information in the "Message" column.


- The → "Start all" option will be selected next before the download operation can be completed with → "Finish".

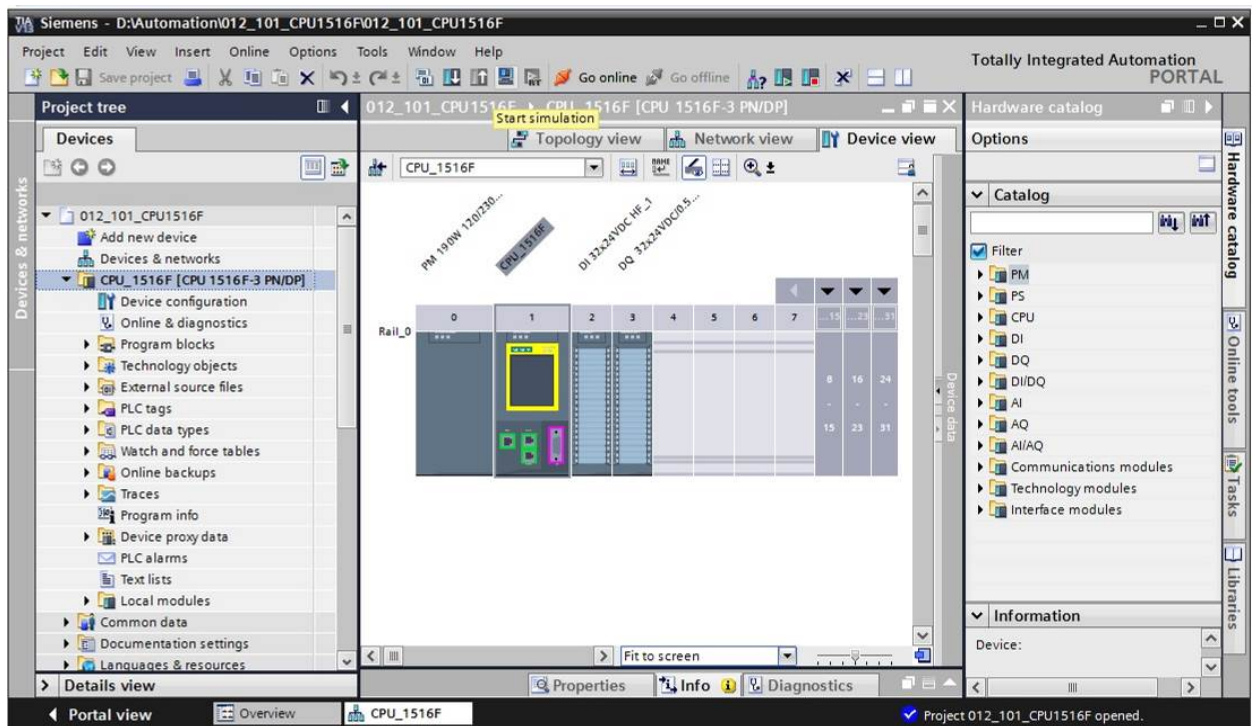


- After a successful download, the project view will open again automatically. A loading report appears in the information field under "General". This can be helpful when troubleshooting an unsuccessful download.



6.12 Download the hardware configuration to the PLCSIM simulation (optional)

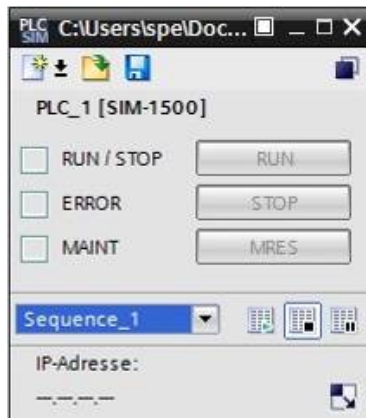
- If no hardware is present, the hardware configuration can **alternatively** be downloaded to a PLC simulation (S7PLCSIM).
- To do so, you must first start the simulation by selecting the → "CPU_1516F [CPU1516F-3 PN/DP]" folder and clicking the  → "Start simulation" icon.



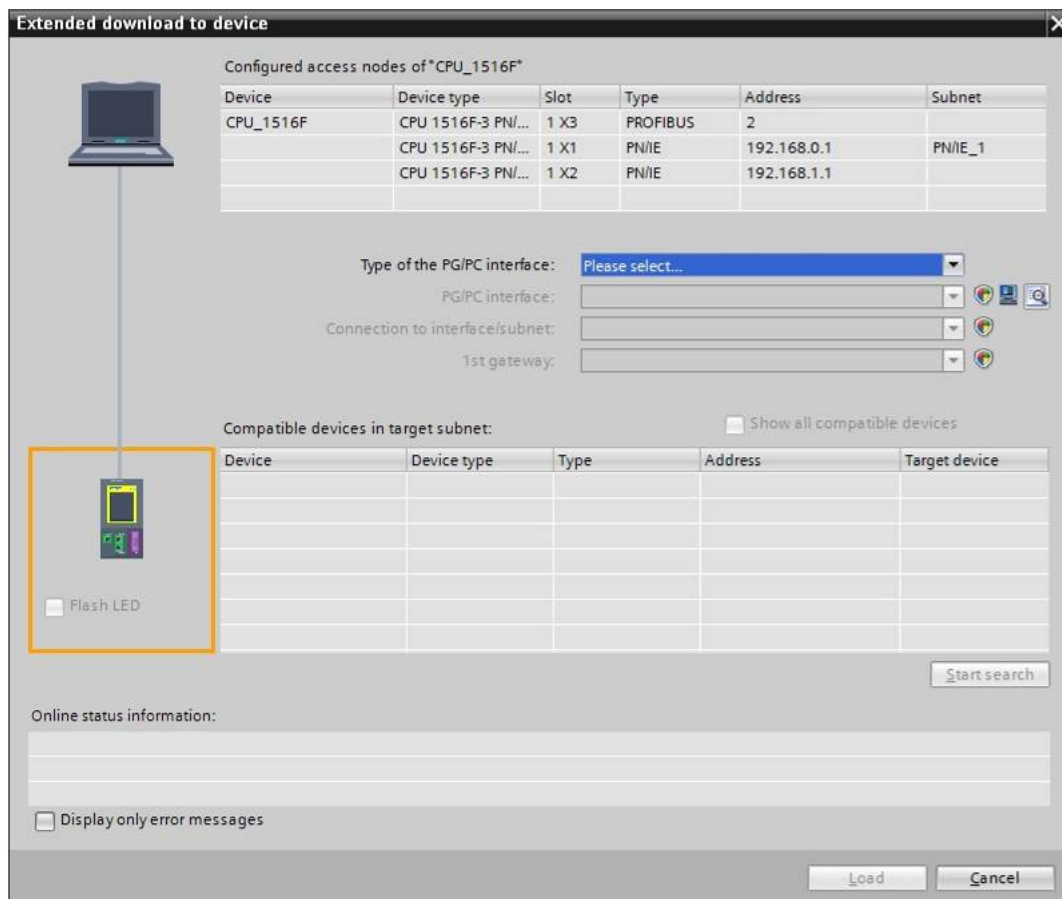
- The prompt that all other online interfaces will be disabled is confirmed with → "OK".



→ The "S7 PLCSIM" software is started in a separate window in the compact view.

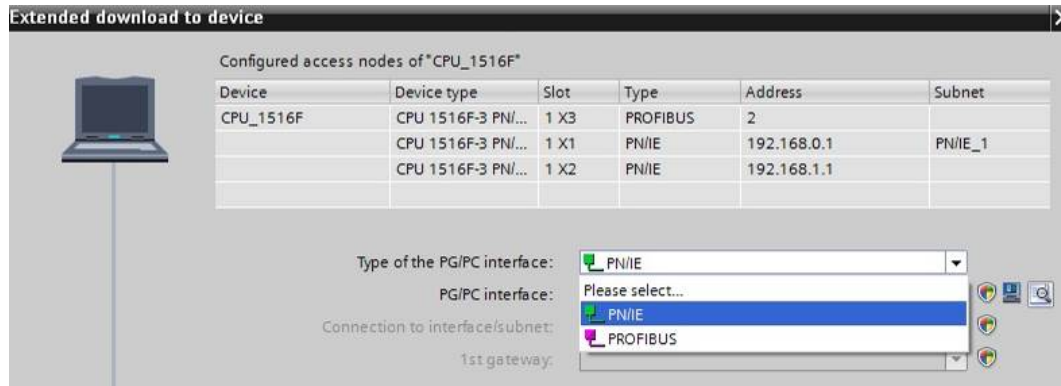


→ The manager for configuring the connection properties (extended download) opens shortly thereafter.

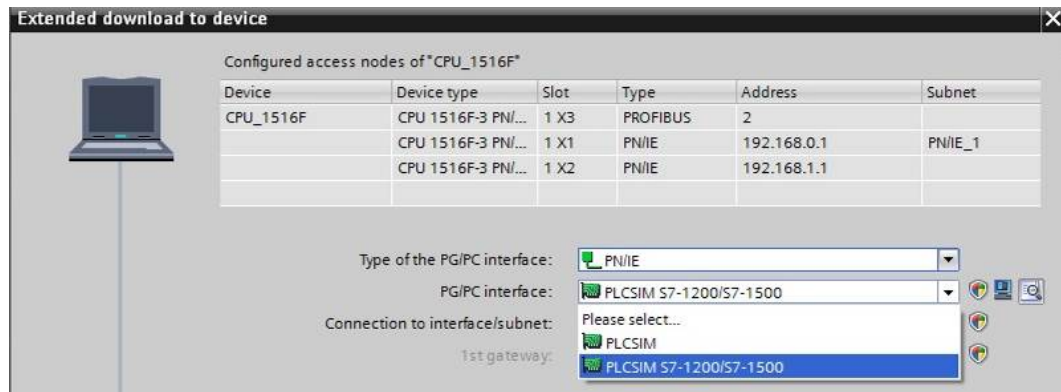


→ First, the interface must be correctly selected. This happens in three steps.

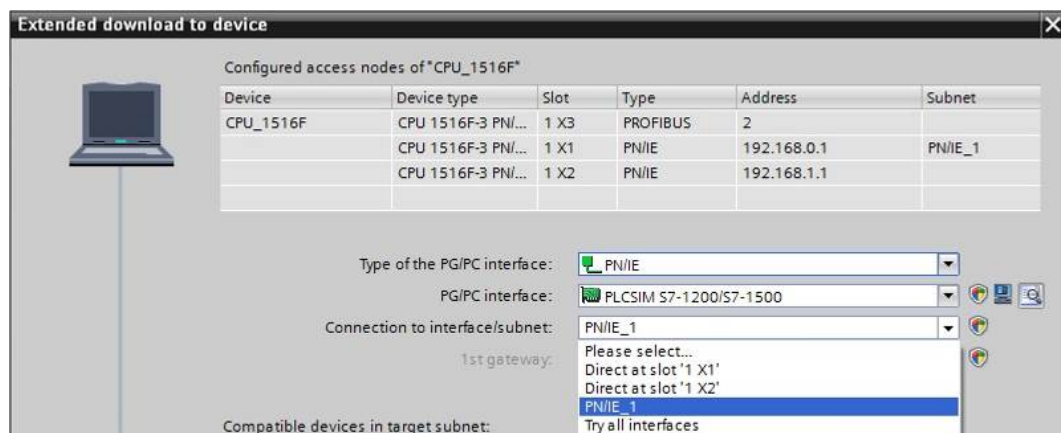
→ Type of the PG/PC interface → PN/IE



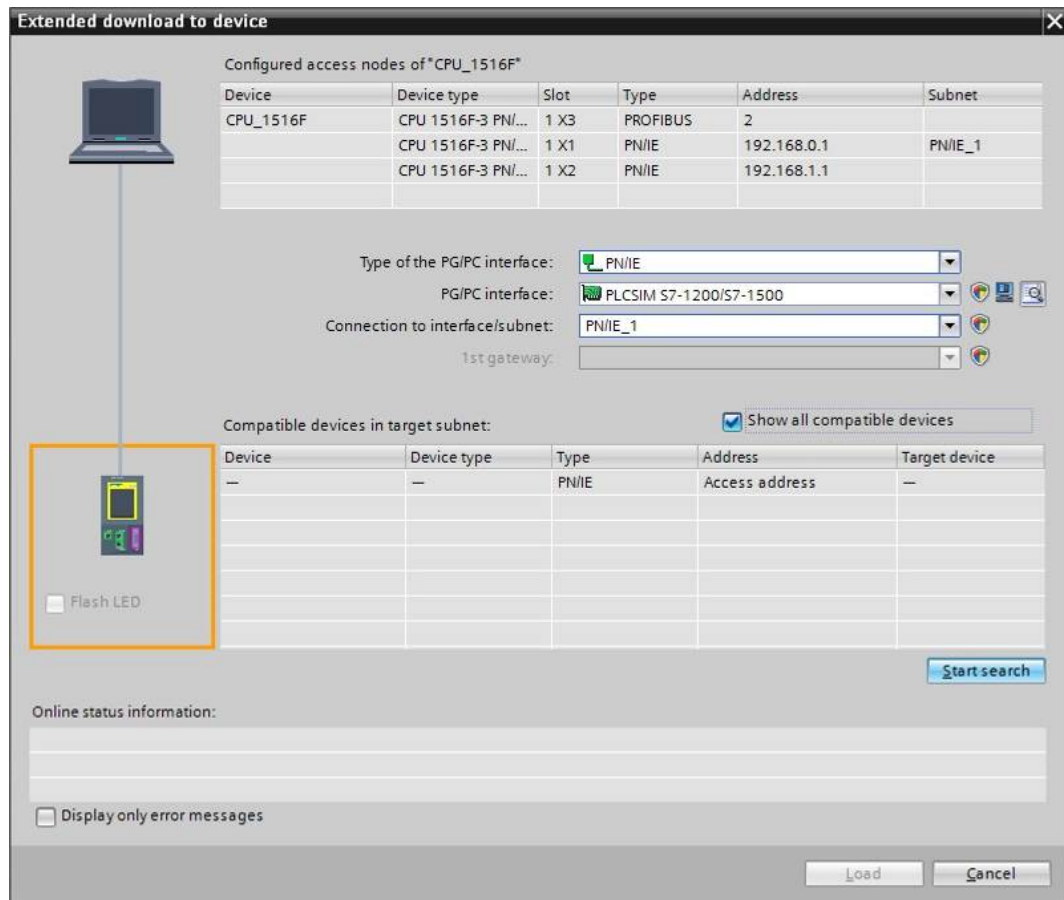
→ PG/PC interface → PLCSIM S7-1200/S7-1500



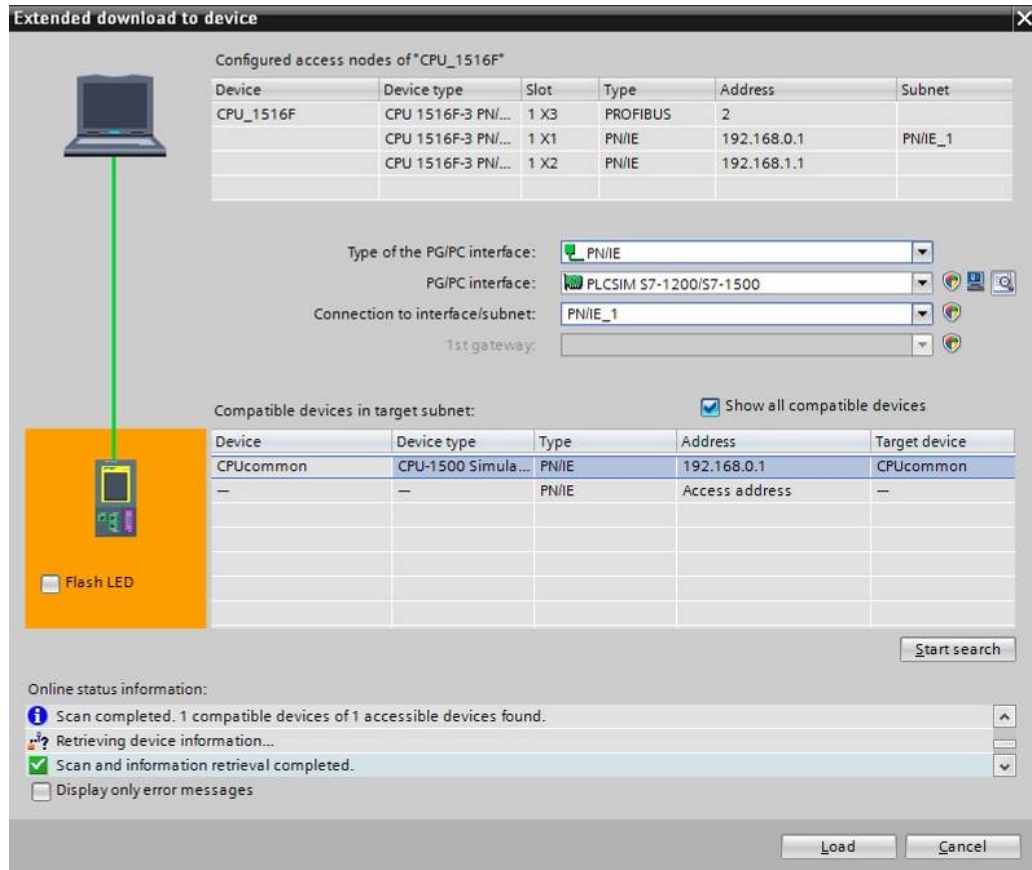
→ Connection to interface/subnet → "PN/IE_1"



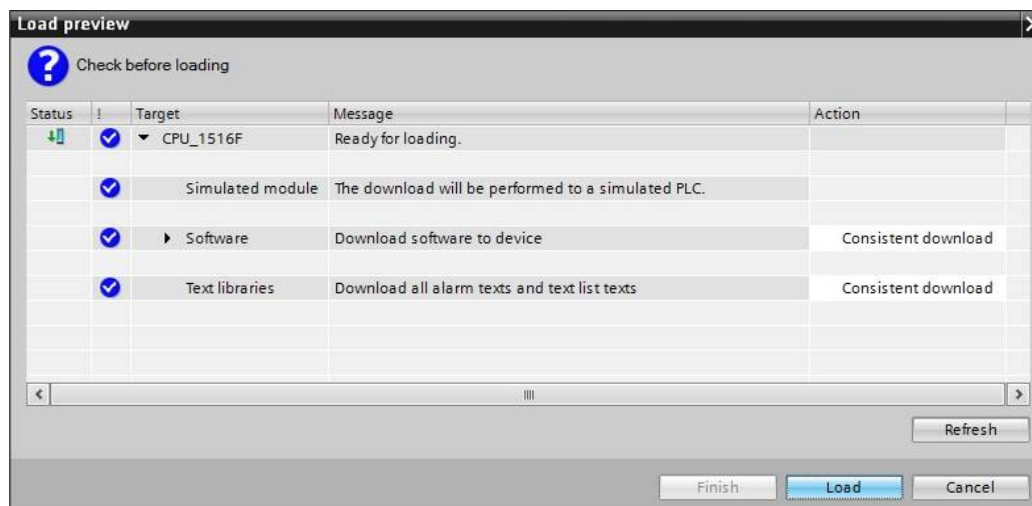
→ The → "Show all compatible devices" check box must be selected. The search for devices in the network is started by clicking the → **Start search** button.



- If the simulation is shown in the "Compatible devices in target subnet" list, it must be selected before the download can be started. (→ "CPU-1500 Simulation" → "Load")

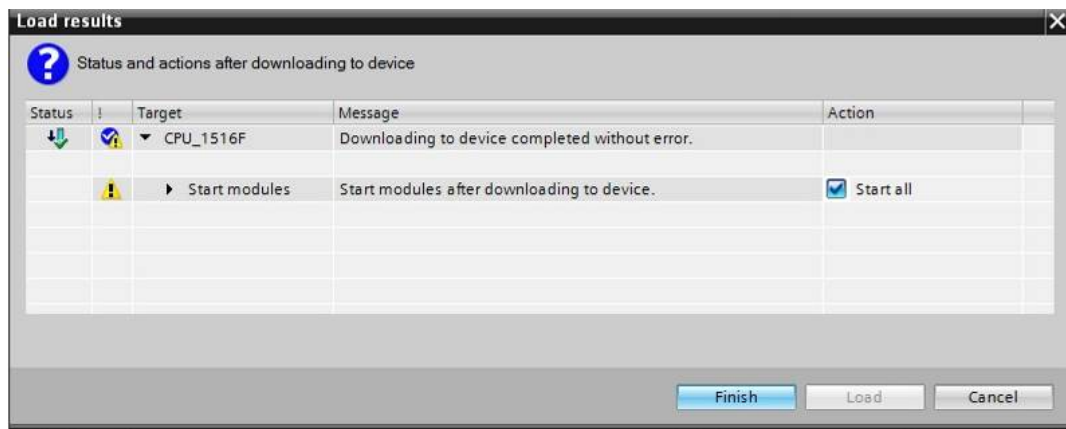


- You first obtain a preview. Confirm the prompt → "Overwrite all" and continue with → "Load".

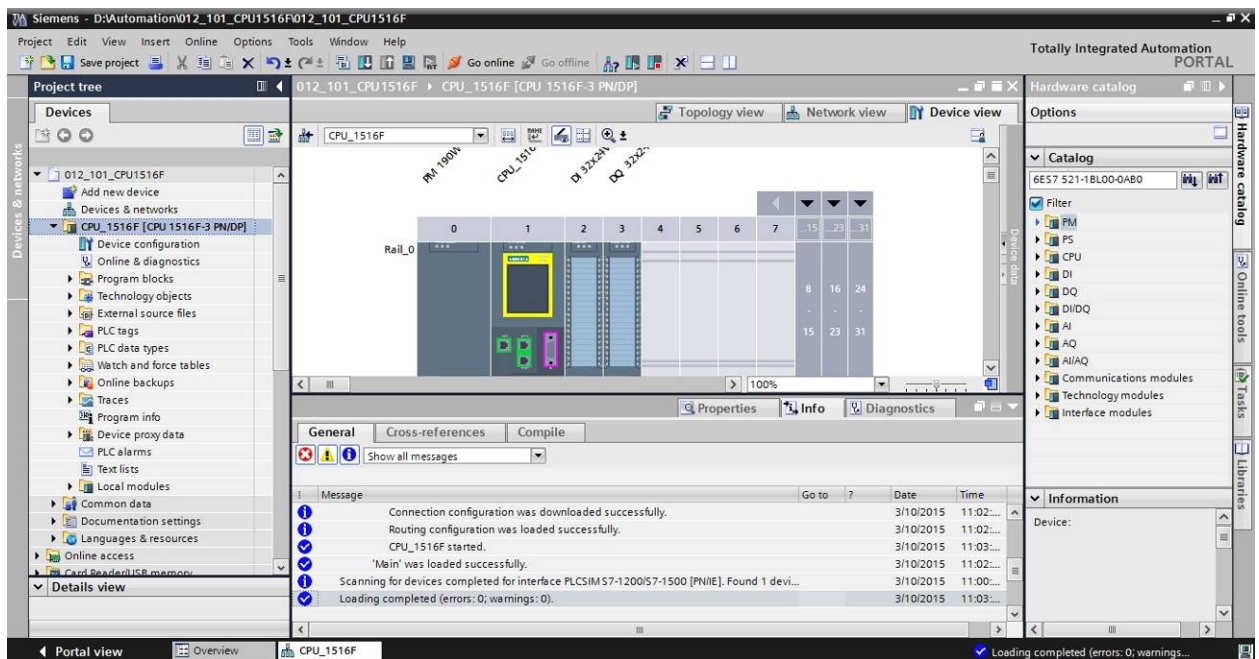



Note: The ✓ symbol should be visible in every line of the "Load preview". You can find additional information in the "Message" column.

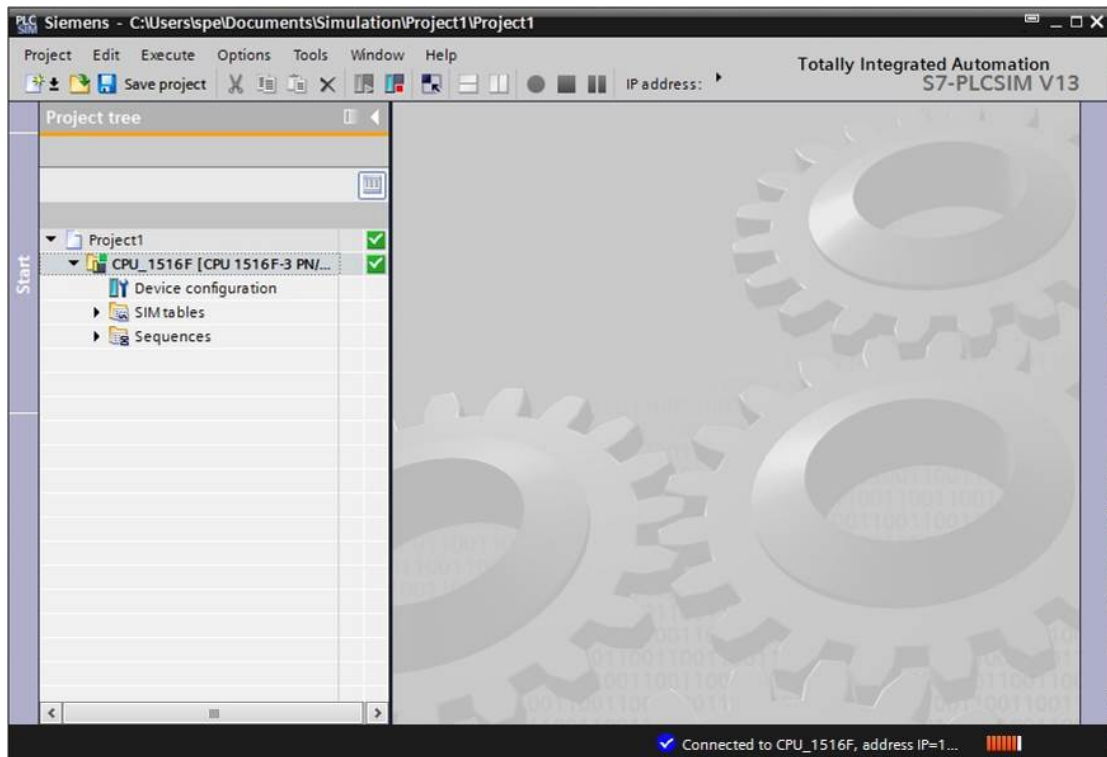
→ The → "Start all" option will be selected next before the download operation can be completed with → "Finish".




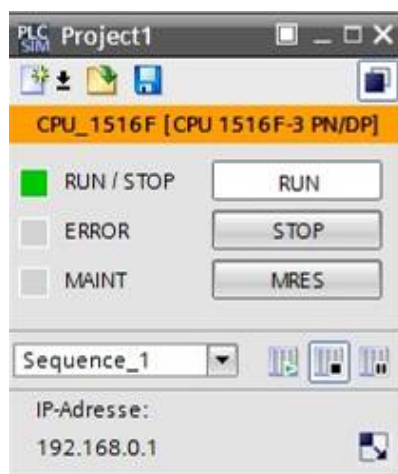
→ After a successful download, the project view will open again automatically. A loading report appears in the information field under "General". This can be helpful when troubleshooting an unsuccessful download.



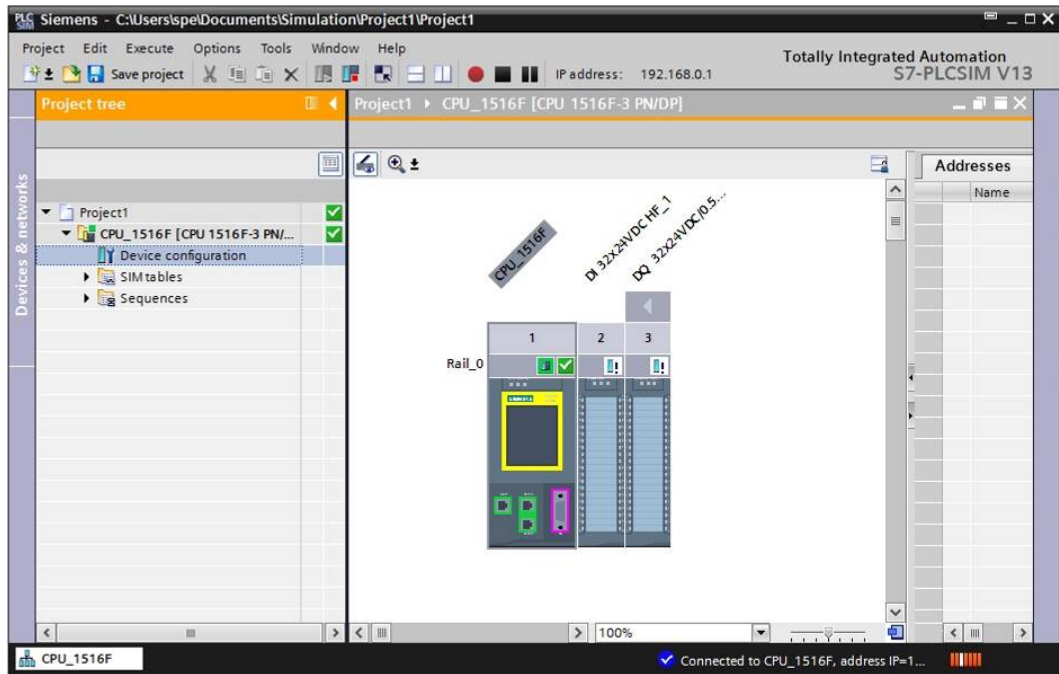
- The PLCSIM simulation has the following appearance in the project view. You can switch to the compact view of the simulation by clicking the →  icon in the menu bar.



- The compact view of the PLCSIM simulation has the following appearance. You can switch back to the project view by clicking the →  icon.



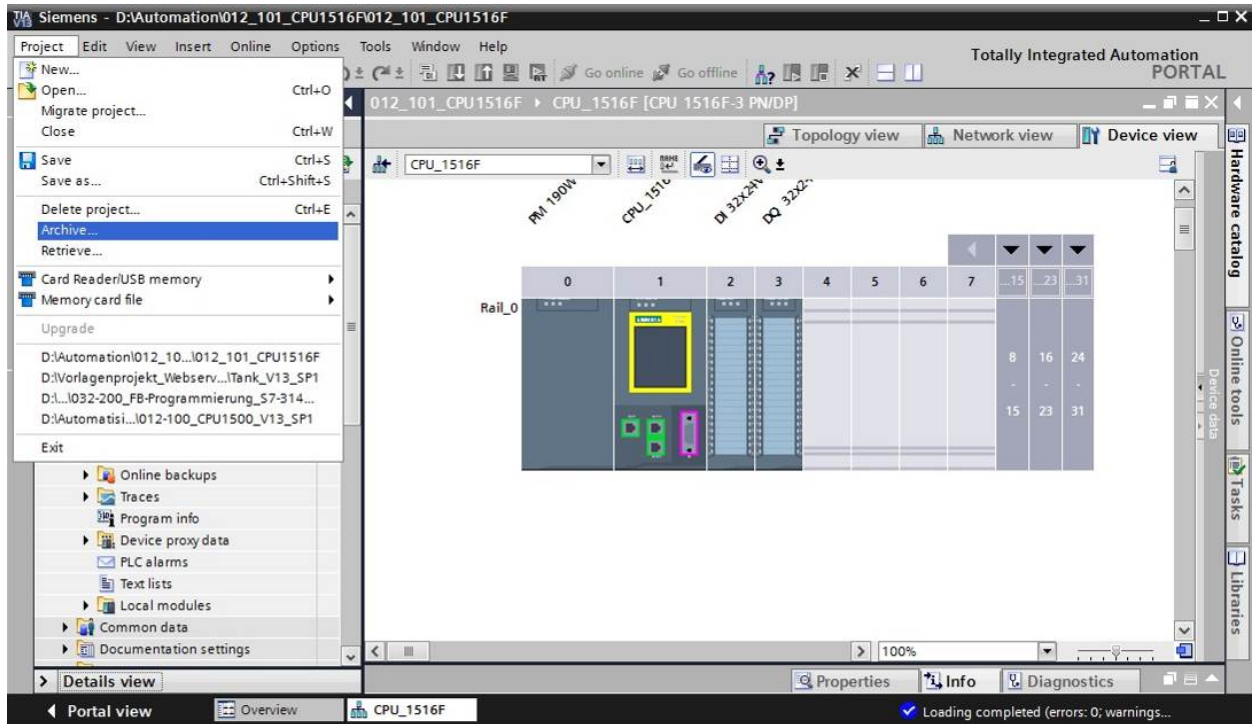
→ You can see the downloaded configuration in the project view by double-clicking → "Device configuration".



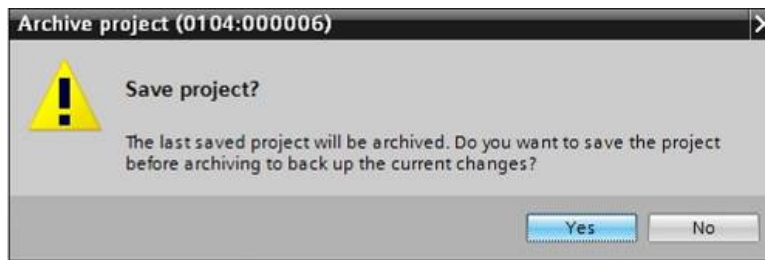
Note: Because this is a simulation, you cannot detect errors in the hardware configuration in this case.

6.13 Archive the project

→ To archive the project, select the → "Archive ..." command in the → "Project" menu.



→ Confirm the prompt to save the project with → "Yes".



→ Select a folder where you want to archive your project and save it as file type "TIA Portal project archive". (→ "TIA Portal project archive" → "SCE_EN_012-101_Hardware configuration_S7-1516F..." → "Save")

6.14 Checklist

No.	Description	Completed
1	Project was created	
2	Slot 0: Power module with correct order number	
3	Slot 1: CPU with correct order number	
4	Slot 1: CPU with correct firmware version	
5	Slot 2: Digital input module with correct order number	
6	Slot 2: Digital input module with correct firmware version	
7	Slot 2: Address area of the digital input module is correct	
8	Slot 3: Digital output module with correct order number	
9	Slot 3: Digital output module with correct firmware version	
10	Slot 3: Address area of the digital output module is correct	
11	Hardware configuration was compiled without error message	
12	Hardware configuration was downloaded without error message	
13	Project was successfully archived	

7 Exercise

7.1 Task – Exercise

The hardware configuration of the SIMATIC CPU 1516F-3 PN/DP Trainer Package is not quite complete. Insert the lacking modules. In so doing, select slot 4 for the analog input module and slot 5 for the analog output module. Use the address area starting from 64 for the analog modules.

- 1X SIMATIC S7-1500, ANALOG INPUT MODULE AI 8 X U/I/RTD/TC, 16BIT RESOLUTION 8 CHANNELS IN GROUPS OF 8 (order number: 6ES7531-7KF00-0AB0)
- 1X SIMATIC S7-1500, ANALOG OUTPUT MODULE AQ 4 X U/I ST, 16BIT RESOLUTION, 4 CHANNELS IN GROUPS OF 4 (order number: 6ES7532-5HD00-0AB0)

7.2 Planning

Plan the implementation of the task on your own.

7.3 Checklist – Exercise

No.	Description	Completed
1	Slot 4: Analog input module with correct order number	
2	Slot 4: Analog input module with correct firmware version	
3	Slot 4: Address area of the analog input module is correct	
4	Slot 5: Analog input module with correct order number	
5	Slot 5: Analog output module with correct firmware version	
6	Slot 5: Address area of the analog output module is correct	
7	Hardware configuration was compiled without error message	
8	Hardware configuration was downloaded without error message	
9	Project was successfully archived	

8 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

3

[illegible]

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC CPU 1512C PN with Software and PM 1507**
Order no.: 6ES7512-1CK00-4AB1
- **SIMATIC CPU 1512C PN with Software, PM 1507 and CP 1542-5 (PROFIBUS)**
Order no.: 6ES7512-1CK00-4AB2
- **SIMATIC CPU 1512C PN with Software**
Order no.: 6ES7512-1CK00-4AB6
- **SIMATIC CPU 1512C PN with Software and CP 1542-5 (PROFIBUS)**
Order no.: 6ES7512-1CK00-4AB7
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Please note that these trainer packages are replaced with successor packages when necessary.

An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, please contact your regional SCE contact

[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and provided to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content are permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG representative: Roland Scheuerer roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, especially Prof. Dr.-Ing. Leon Urbas and Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

TABLE OF CONTENTS

1	Goal	5
2	Prerequisite.....	5
3	Theory.....	5
3.1	SIMATIC S7-1500 automation system	5
3.1.1	Range of modules	7
3.1.2	Example configuration	10
3.2	Operator control and display elements of the CPU 1512C-1 PN	11
3.2.1	Front view of the CPU 1512C-1 PN with integrated display.....	11
3.2.2	Status and error displays.....	12
3.2.3	Operator control and connection elements of the CPU 1512C-1 PN behind the hinged front cover	13
3.2.4	SIMATIC memory card	14
3.2.5	Mode switch.....	14
3.2.6	Display of the CPU	15
3.3	Memory areas of the CPU 1512C-1 PN and the SIMATIC memory card	17
3.4	STEP 7 Professional V13 (TIA Portal V13) programming software	18
3.4.1	Project	19
3.4.2	Hardware configuration	19
3.4.3	Centralized and decentralized automation structure.....	20
3.4.4	Planning the hardware.....	20
3.4.5	TIA Portal – Project view and portal view.....	21
3.4.6	Basic settings for the TIA Portal	23
3.4.7	Setting the IP address on the programming device	25
3.4.8	Setting the IP address in the CPU.....	28
3.4.9	Formatting the memory card in the CPU.....	31
3.4.10	Resetting the CPU to factory settings	32

4	Task	33
5	Planning	33
6	Structured step-by-step instructions	34
6.1	Create a new project	34
6.2	Inserting the CPU 1512C-1 PN	35
6.3	Configure Ethernet interface of the CPU 1512C-1 PN	39
6.4	Insert power module PM 190W 120/230VAC	41
6.5	Configure the address areas for the digital and analog inputs and outputs	42
6.6	Save and compile the hardware configuration	43
6.7	Download the hardware configuration to the device	45
6.8	Download the hardware configuration to the PLCSIM simulation (optional)	50
6.9	Archive the project	57
6.10	Checklist	58
7	Exercise	59
7.1	Task – Exercise	59
7.2	Planning	59
7.3	Checklist – Exercise	60
8	Additional information	61

SPECIFIED HARDWARE CONFIGURATION – SIMATIC S7-1512C-1 PN

1 Goal

In this chapter, you will first learn how to **create a project**. You are then shown how the **hardware is configured**.

2 Prerequisite

You do not need any previous knowledge from other chapters to successfully complete this chapter.

3 Theory

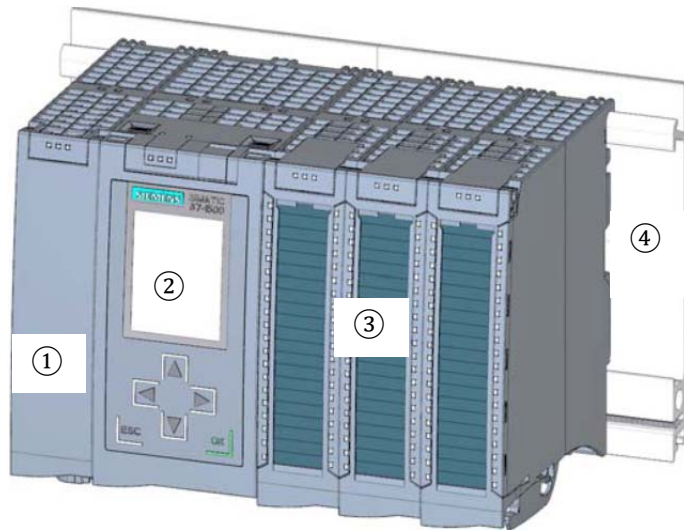
3.1 SIMATIC S7-1500 automation system

The SIMATIC S7-1500 automation system is a modular controller system for the middle to upper performance range. A comprehensive range of modules is available to optimally adapt the system to the automation task.

SIMATIC S7-1500 is the next generation of the SIMATIC S7-300 and S7-400 automation systems with the following new performance features.

- Increased system performance
- Integrated motion control functionality
- PROFINET IO IRT
- Integrated display for machine-level operation and diagnostics
- STEP 7 language innovations while maintaining proven functions

The S7-1500 controller consists of a power supply ①, a CPU with integrated display ② and, in the case of compact CPUs, integrated inputs and outputs. In addition, there are input and output modules for digital and analog signals ③. If necessary, communication processors and function modules are also used for special tasks such as PROFIBUS communication or stepper motor control. The up to 32 modules are mounted on a mounting rail with integrated DIN rail profile ④.



The programmable logic controller (PLC) uses the S7 program to monitor and control a machine or process. In doing so, the S7 program scans the I/O modules via input addresses (%I) and addresses their output addresses (%Q).

The system is programmed with the STEP 7 Professional V13 software.

3.1.1 Range of modules

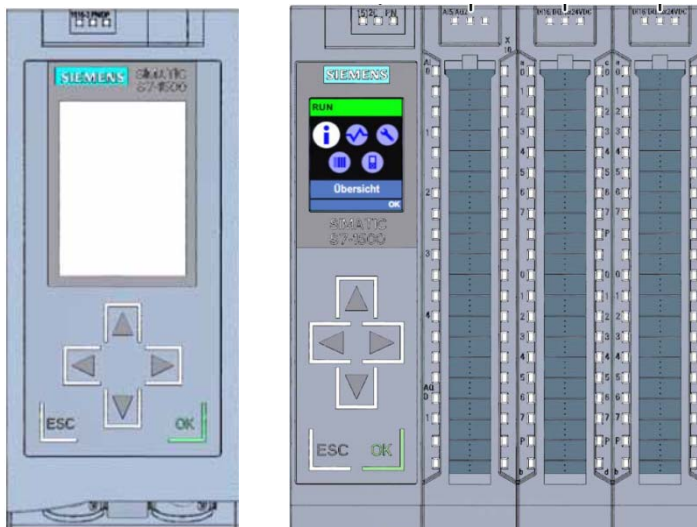
The SIMATIC S7-1500 is a modular automation system and offers the following range of modules:

Central processing units (CPUs) with integrated display

The CPUs have different performance capability and execute the user program. In addition, the other modules are supplied power via the backplane bus with the integrated system power supply.

Additional properties and functions of the CPU:

- Communication via Ethernet
- Communication via PROFIBUS/PROFINET
- HMI communication for HMI devices
- Web server
- Integrated technology functions (e.g. PID controller, motion control, etc.)
- System diagnostics
- Integrated security (e.g. know-how, copy, access, integrity protection)
- Integrated digital and analog inputs and outputs (for compact CPUs)



System power supply modules (PS) (rated input voltages 24 V DC to 230 V AC/DC)

with connection to the backplane bus supply the configured modules with the internal supply voltage.

**Load current supply modules (PM) (rated input voltages 120/230 V AC)**

do not have a connection to the backplane bus of the S7-1500 automation system. The load current supply is used to supply 24 V DC to the system power supply of the CPU, the input and output circuits of I/O modules and the sensors and actuators.



I/O modules

for digital input (DI) / digital output (DQ) / analog input (AI) / analog output (AQ)



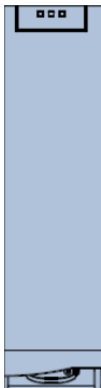
4

Technology modules (TM)

as incremental encoders and pulse encoders with/without direction signal.

**Communication modules (CM)**

for serial communication RS232 / RS422 / RS485, PROFIBUS and PROFINET.



SIMATIC memory card

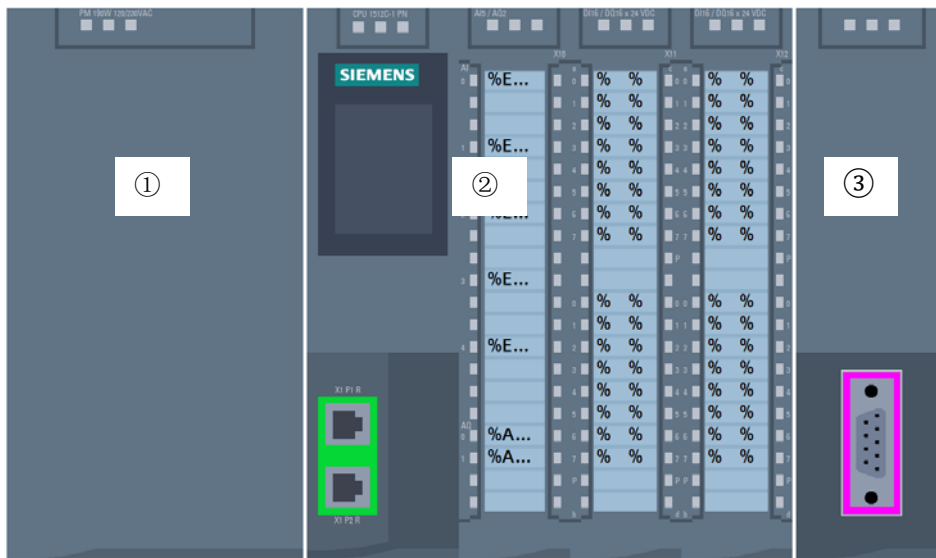
up to a maximum of 32 GB for storing program data and for easy replacement of CPUs during maintenance.



4

3.1.2 Example configuration

The following configuration of an S7-1500 automation system will be used for the program example in this curriculum.



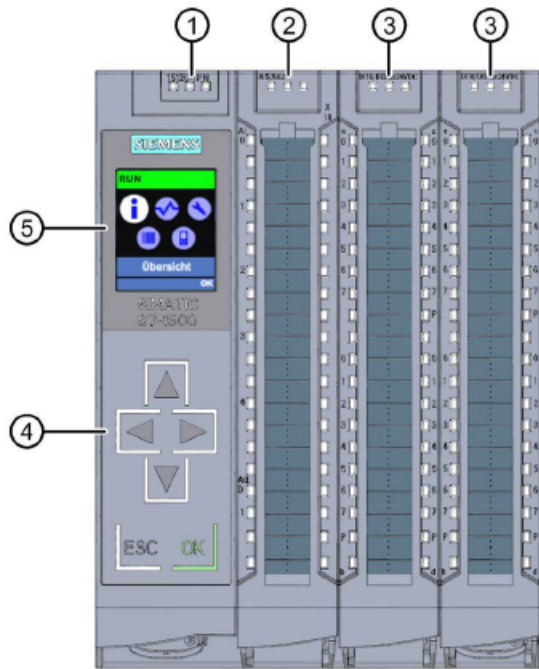
- ① Load current supply module PM 1507 with input 120/230 V AC, 50 Hz / 60 Hz, 190 W and output 24 V DC / 8 A
- ② Central module compact CPU 1512C-1 PN with integrated PROFINET interfaces
- ③ Communications processor CP 1542-5 for connection to PROFIBUS DP

3.2 Operator control and display elements of the CPU 1512C-1 PN

The figure below shows the operator control and display elements of a CPU 1512C-1 PN.

The arrangement and number of elements differ from this figure for other CPUs.

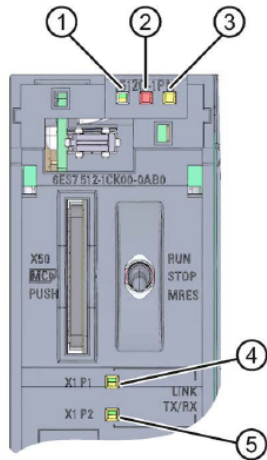
3.2.1 Front view of the CPU 1512C-1 PN with integrated display



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Status and error displays RUN/ERROR of the analog on-board I/O
- ③ Status and error displays RUN/ERROR of the digital on-board I/O
- ④ Control keys
- ⑤ Display

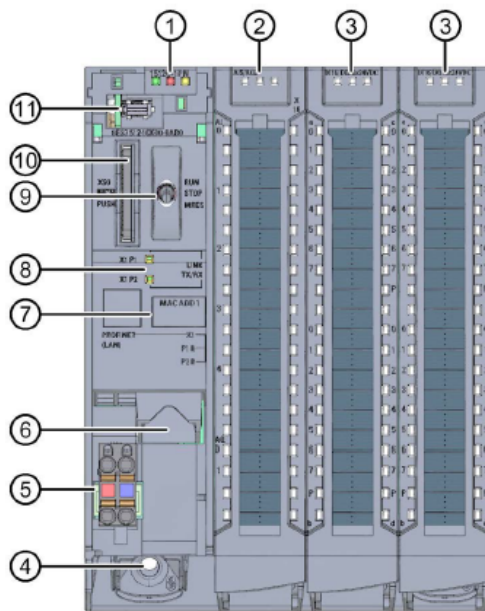
3.2.2 Status and error displays

The CPU comes with the following LED displays:



- ① RUN/STOP LED (yellow/green LED)
- ② ERROR LED (red LED)
- ③ MAINT LED (yellow LED)
- ④ LINK RX/TX LED for port X1 P1 (yellow/green LED)
- ⑤ LINK RX/TX LED for port X1 P2 (yellow/green LED)

3.2.3 Operator control and connection elements of the CPU 1512C-1 PN behind the hinged front cover



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Status and error displays RUN/ERROR of the analog on-board I/O
- ③ Status and error displays RUN/ERROR of the digital on-board I/O
- ④ Fastening screw
- ⑤ Connection for supply voltage
- ⑥ PROFINET interface (X1) with 2 ports (X1 P1 and X1 P2)
- ⑦ MAC address
- ⑧ LED displays for the 2 ports (X1 P1 and X1 P2) of the PROFINET interface X1
- ⑨ Mode switch
- ⑩ Slot for the SIMATIC memory card
- ⑪ Display connection

Note: The hinged front cover with the display can be removed and inserted during operation.

3.2.4 SIMATIC memory card

A SIMATIC Micro Memory Card is used as the memory module for the CPUs. This is a preformatted memory card that is compatible with the Windows file system. It is available with various storage capacities and can be used for the following purposes:

- Transportable data storage medium
- Program card
- Firmware update card

The MMC **must** be inserted to operate the CPU as the CPUs have no integrated load memory. A commercially available SD card reader is needed to write/read the SIMATIC memory card with the programming device or PC. This allows files to be copied directly to the SIMATIC memory card using Windows Explorer, for example.

Note: It is recommended that the SIMATIC memory card only be removed or inserted when the CPU is in the POWER OFF state.

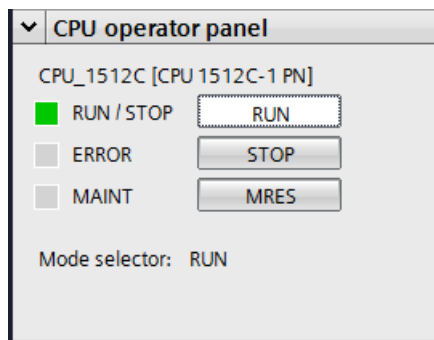
3.2.5 Mode switch

The mode switch allows you to set the operating mode of the CPU.
The mode switch is designed as a toggle switch with 3 switch positions.

Position	Meaning	Explanation
RUN	RUN mode	The CPU is executing the user program.
STOP	STOP mode	The CPU is not executing the user program.
MRES	Memory reset	Position for CPU memory reset.

You can also use the button on the CPU operator panel of the STEP 7 Professional V13 software in Online & Diagnostics to switch the operating mode (**STOP** or **RUN**).

The operator panel also contains an **MRES** button for performing a memory reset and displays the status LEDs of the CPU.



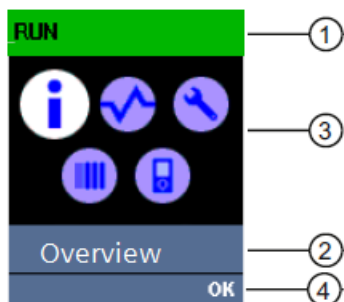
3.2.6 Display of the CPU

The S7-1500 CPU has a hinged front cover with a display and control keys. Control data and status data can be displayed in various menus on the display and numerous settings can be made. You use the control keys to navigate through the menus.

The display of the CPU offers the following functions:

- 6 different display languages can be selected.
- Diagnostic messages are displayed in plain text.
- The interface settings can be changed locally.
- Password assignment for display operation is possible through the TIA Portal.

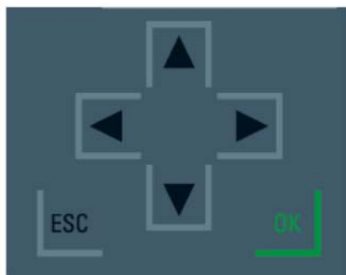
View of the display of an S7-1500:



- ① CPU status information
- ② Submenu name
- ③ Information display field
- ④ Navigation aid, e.g. OK/ESC or the page number

Control keys of the display

- Four arrow keys: "up", "down", "left", "right"
- An ESC key
- An OK key



Functions of the "OK" and "ESC" keys






→ For menu commands in which an input can be made:

- OK → valid access to the menu command, confirmation of input and exit from editing mode
- ESC → restoration of original content (which means changes are not saved) and exit from editing mode

→ For menu commands in which no input can be made:

- OK → to next submenu command
- ESC → back to previous menu command

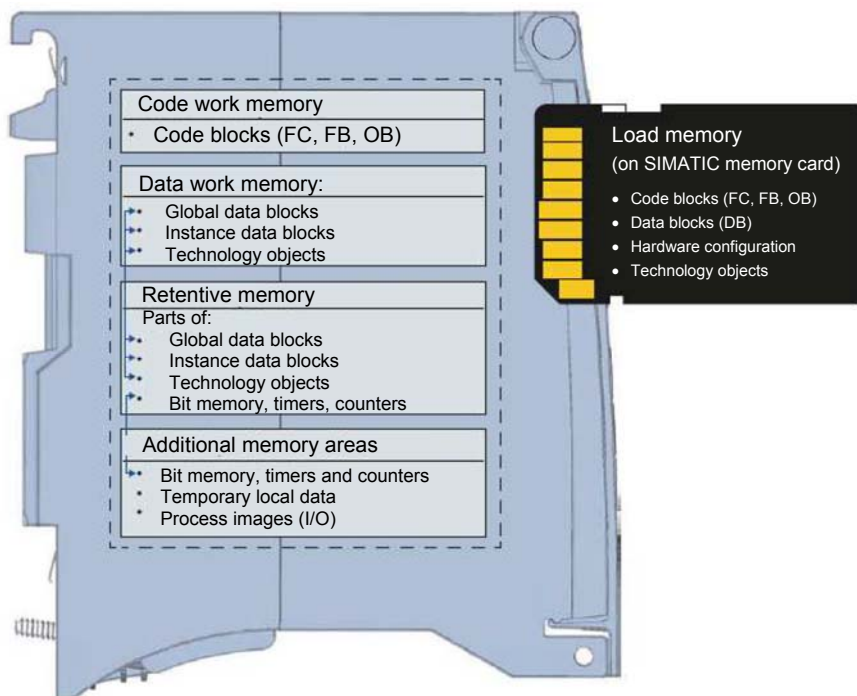
Available submenus of the display:

Main menu commands	Meaning	Explanation
	Overview	The "Overview" menu contains information about the properties of the CPU.
	Diagnostics	The "Diagnostics" menu contains information about diagnostic messages, the diagnostic description and the indication of interrupts. There is also information about the network properties of each interface of the CPU.
	Settings	In the "Settings" menu, IP addresses of the CPU are assigned, the date, time of day, time zones, operating modes (RUN/STOP) and protection levels are set, a memory reset and reset to factory settings of the CPU can be performed and the status of firmware updates is displayed.
	Modules	The "Modules" menu contains information about the modules that are used in your configuration. The modules can be used as central or distributed modules. Distributed modules are connected to the CPU via PROFINET and/or PROFIBUS. You have the option here to set the IP addresses for a CP.
	Display	In the "Display" menu, settings are made for all aspects of the display, such as the language setting, brightness setting and Energy-saving mode. (Energy-saving mode darkens the display. Standby mode switches off the display.)

3.3 Memory areas of the CPU 1512C-1 PN and the SIMATIC memory card

The following figure shows the memory areas of the CPU and the load memory on the SIMATIC memory card.

In addition to the load memory, other data can be loaded onto the SIMATIC memory card using Windows Explorer. This includes recipes, data logs, project backups and additional documentation for the program.



Load memory

Load memory is non-volatile memory for code blocks, data blocks, technology objects and the hardware configuration. When these objects are downloaded to the CPU, they are first stored in the load memory. This memory is located on the SIMATIC memory card.

Work memory

Work memory is volatile memory that contains the code and data blocks. The work memory is integrated into the CPU and cannot be expanded. In S7-1500 CPUs, the work memory is divided into two areas:

- Code work memory:
 - The code work memory contains runtime-relevant parts of the program code.
- Data work memory:
 - The data work memory contains the runtime-relevant parts of the data blocks and technology objects.

At the operating mode transitions from POWER ON to startup and from STOP to startup, tags of global data blocks, instance data blocks and technology objects are initialized with their start values. Retentive tags retain their actual values that were saved in the retentive memory.

Retentive memory

Retentive memory is non-volatile memory for saving certain data in the event of power failure. The tags and operand areas that have been defined as retentive are saved in the retentive memory. This data is retained beyond power-off or power failure.

All other program tags are set to their start values at the operating mode transitions from POWER ON to startup and from STOP to startup.

The content of retentive memory is deleted by the following actions:

- Memory reset
- Reset to factory settings

Note: Certain tags of technology objects are also stored in the retentive memory. These tags are not deleted by a memory reset.

3.4 STEP 7 Professional V13 (TIA Portal V13) programming software

STEP 7 Professional V13 (TIA Portal V13) software is the programming tool for the following automation systems:

- SIMATIC S7-1500
- SIMATIC S7-1200
- SIMATIC S7-300
- SIMATIC S7-400
- SIMATIC WinAC

STEP 7 Professional V13 provides the following functions for plant automation:

- Configuration and parameter assignment of the hardware
- Specification of the communication
- Programming
- Testing, commissioning and servicing with operational/diagnostic functions
- Documentation
- Creation of visualizations for SIMATIC Basic Panels using the integrated WinCC Basic software
- Visualization solutions for PCs and other panels can also be created with other WinCC software packages

Support is provided for all functions through detailed online help.

3.4.1 Project

To implement a solution for an automation and visualization task, you create a project in the TIA Portal. A project in the TIA Portal contains the configuration data for the configuration and networking of devices as well as the programs and the configuration of the visualization.

3.4.2 Hardware configuration

The *hardware configuration* includes the configuration of the devices, consisting of the hardware of the automation systems, the intelligent field devices and the hardware for visualization. The configuration of the networks specifies the communication between the various hardware components. The individual hardware components are *inserted in the hardware configuration* from catalogs.

The hardware of automation systems comprises controllers (CPUs), signal modules for input and output signals (SMs) and communication and interface modules (CP, IM). Other power supply and current supply modules (PS, PM) are also available to supply the modules.

The signal modules and intelligent field devices connect the input and output data of the process to be automated and visualized to the automation system.

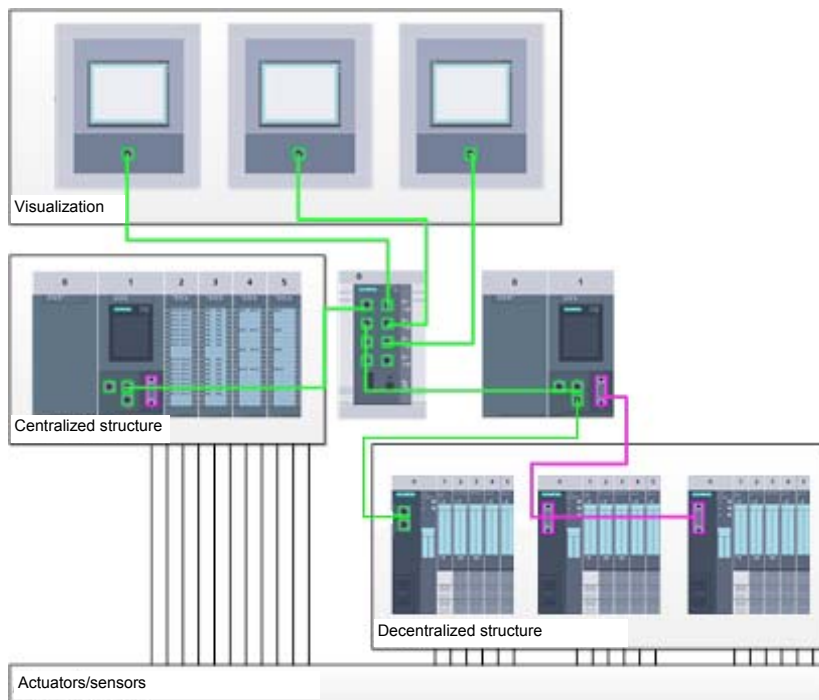


Figure 1: Example of hardware configuration with centralized and decentralized structures

The hardware configuration enables the downloading of automation and visualization solutions to the automation system and access to the connected signal modules by the controller.

3.4.3 Centralized and decentralized automation structure

Figure 1 shows an automation structure that contains both centralized and decentralized structures.

In centralized structures, the input and output signals of the process are transmitted by way of conventional wiring to the signal modules, which are connected directly to the controller. Conventional wiring refers to the connection of sensors and actuators using 2-wire or 4-wire cables.

The decentralized structure is the predominant structure used today. Here, the sensors and actuators are wired conventionally only as far as the signal modules of the field devices. The signal transmission from the field devices to the controller is implemented using an industrial communication system.

Both classic fieldbuses such as PROFIBUS, Modbus and Foundation Fieldbus as well as Ethernet-based communication systems such as PROFINET can be used as the industrial communication system.

In addition, intelligent field devices in which stand-alone programs run can also be connected via the communication system. These programs can also be created with the TIA Portal.

3.4.4 Planning the hardware

Before you can configure the hardware, you must plan it (hardware planning). In general, you begin by selecting which controllers are needed and how many. Next you select the communication modules and signal modules. The selection of signal modules is based on the number and type of inputs and outputs needed. As the final step, a power supply that ensures that the necessary power is supplied must be selected for each controller or field device.

The functionality required and the ambient conditions are of vital importance for planning the hardware configuration. For example, the temperature range in the application area sometimes limits the devices available for selection. Fail-safe operation might be another requirement, for example.

The [TIA Selection Tool](#) (Select automation technology → TIA Selection Tool and follow the instructions) provides you support.

Note: TIA Selection Tool requires Java.

Note for online research: If more than one manual is available, you should look for the description "Device Manual", "Product Manual" or simply "Manual" (as opposed to "Function Manual", "List Manual", "System Manual", etc.) in order to find the device specifications.

3.4.5 TIA Portal – Project view and portal view

The TIA Portal has two important views. When started, the portal view appears by default. This view makes getting started easier, especially for beginning users.

The portal view provides a task-oriented view of the tools for working on the project. Here, you can quickly decide what you want to do and open the tool for the task at hand. If necessary, a change to the project view takes place automatically for the selected task.

Figure 2 shows the portal view. At the bottom left, there is an option to switch between this view and the project view.

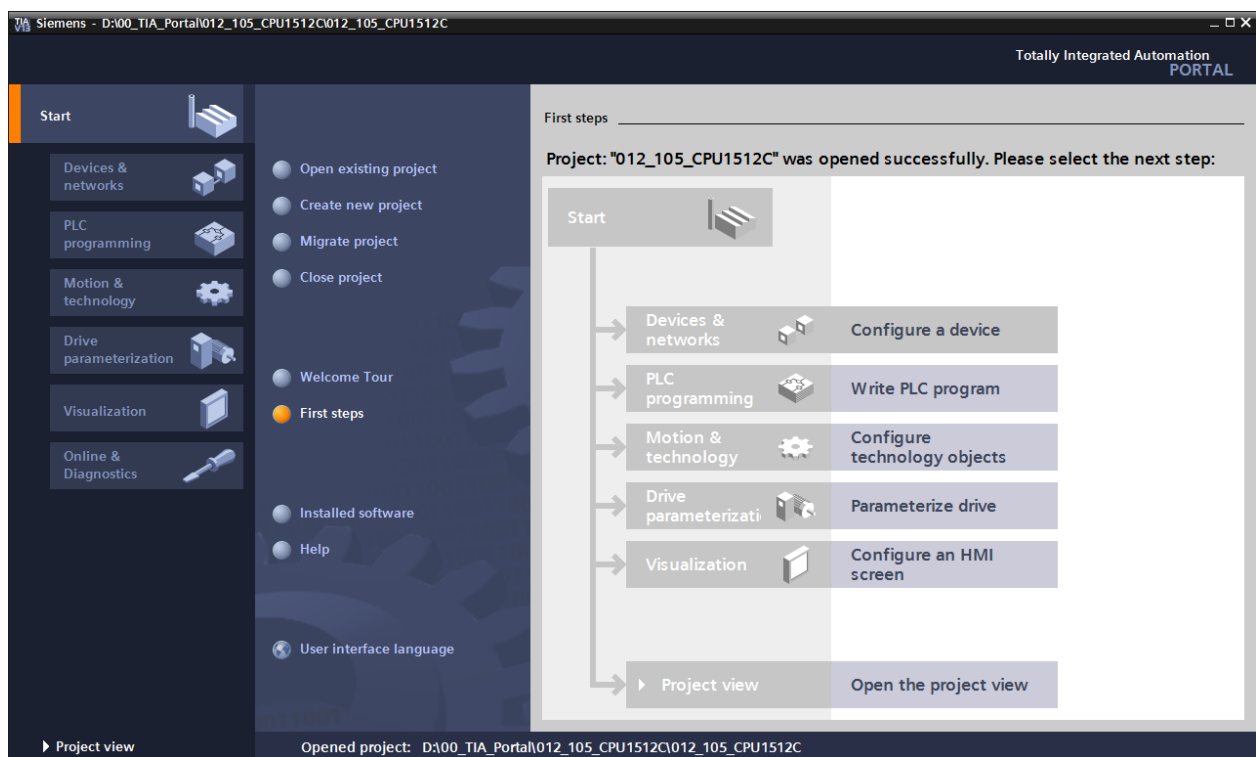


Figure 2: Portal view

The project view, as shown in Figure 3, is used for hardware configuration, programming, creation of the visualization and many other tasks.

By default, the project view displays the menu bar with the toolbars at the top, the project tree with all components of a project on the left and the so-called task cards with instructions and libraries, for example, on the right.

If an element (for example, the device configuration) is selected in the project tree, it is displayed in the center and can be worked on there.

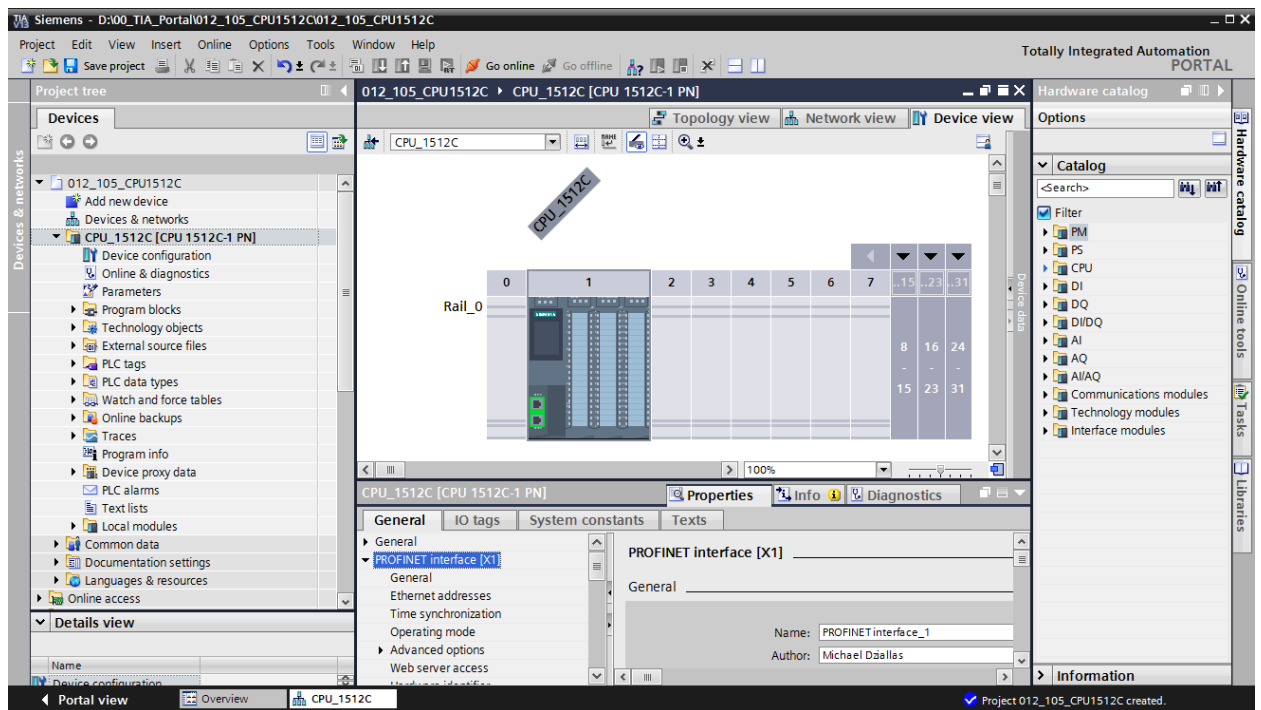
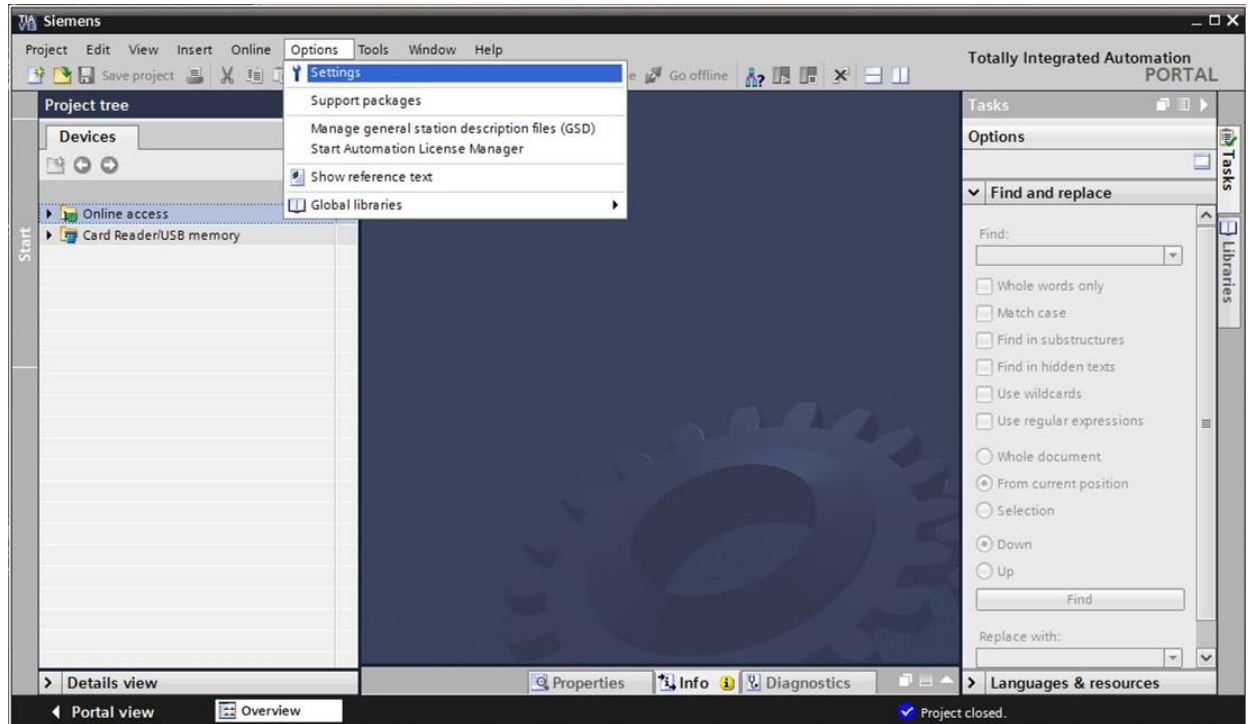


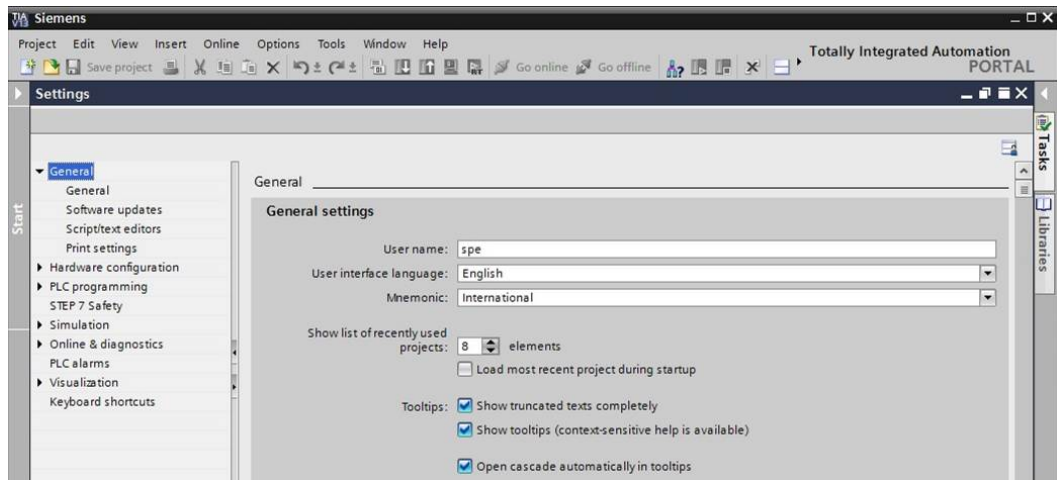
Figure 3: Project view

3.4.6 Basic settings for the TIA Portal

- Users can specify their own default settings for certain settings in the TIA Portal. A few important settings are shown here.
- In the project view, select the →"Options" menu and then → "Settings".



- One basic setting is the selection of the user interface language and the language for the program display. In the curriculums to follow, "English" will be used for both settings.
- Under → "General" in "Settings", select "User interface language → English" and "Mnemonic → International".




Note: These settings can always be changed.

3.4.7 Setting the IP address on the programming device

To program SIMATIC S7-1500 from the PC, the programming device or a laptop, you need a TCP/IP connection or an optional PROFIBUS connection.

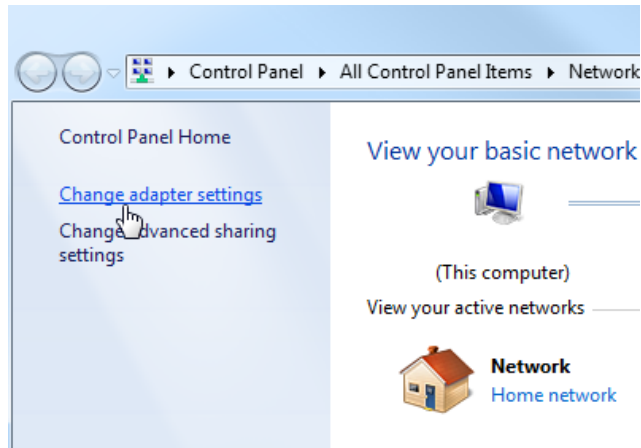
For the PC and SIMATIC S7-1500 to communicate with each other via TCP/IP, it is important that the IP addresses of both devices match.

First, we show you how to set the IP address of a computer with the Windows 7 operating system.

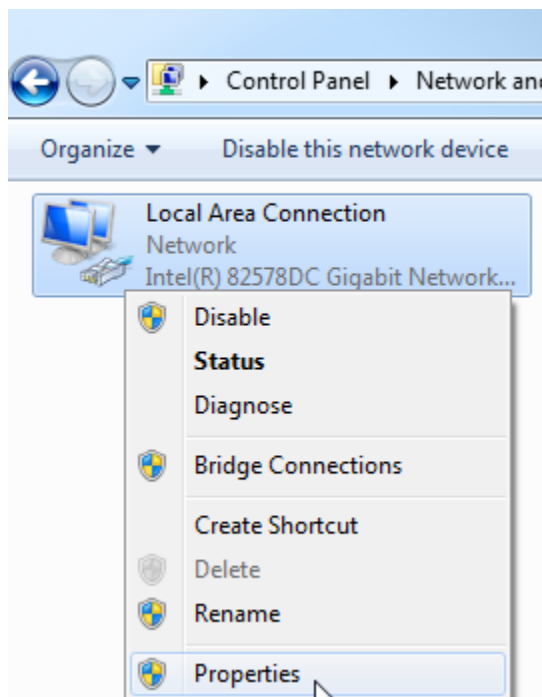
→ Locate the network icon in the taskbar at the bottom  and click → "Open Network and Sharing Center".



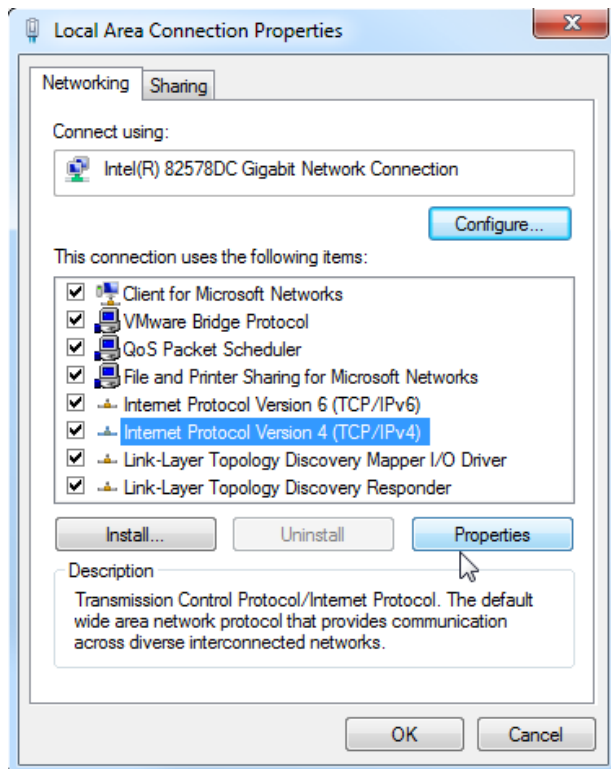
→ In the open Network and Sharing Center window, click → "Change adapter settings".



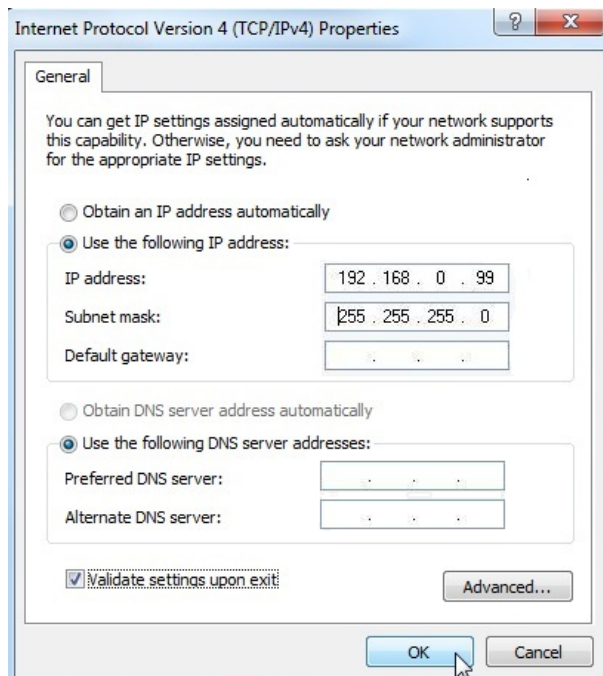
→ Select the desired → "Local Area Connection" that you want to use to connect to the controller and click → "Properties".



→ Next, select → "Properties" for → "Internet Protocol Version 4 (TCP/IP)".



→ You can use the following IP address, for example → IP address: 192.168.0.99 → Subnet mask 255.255.255.0 and accept the settings (→ "OK")



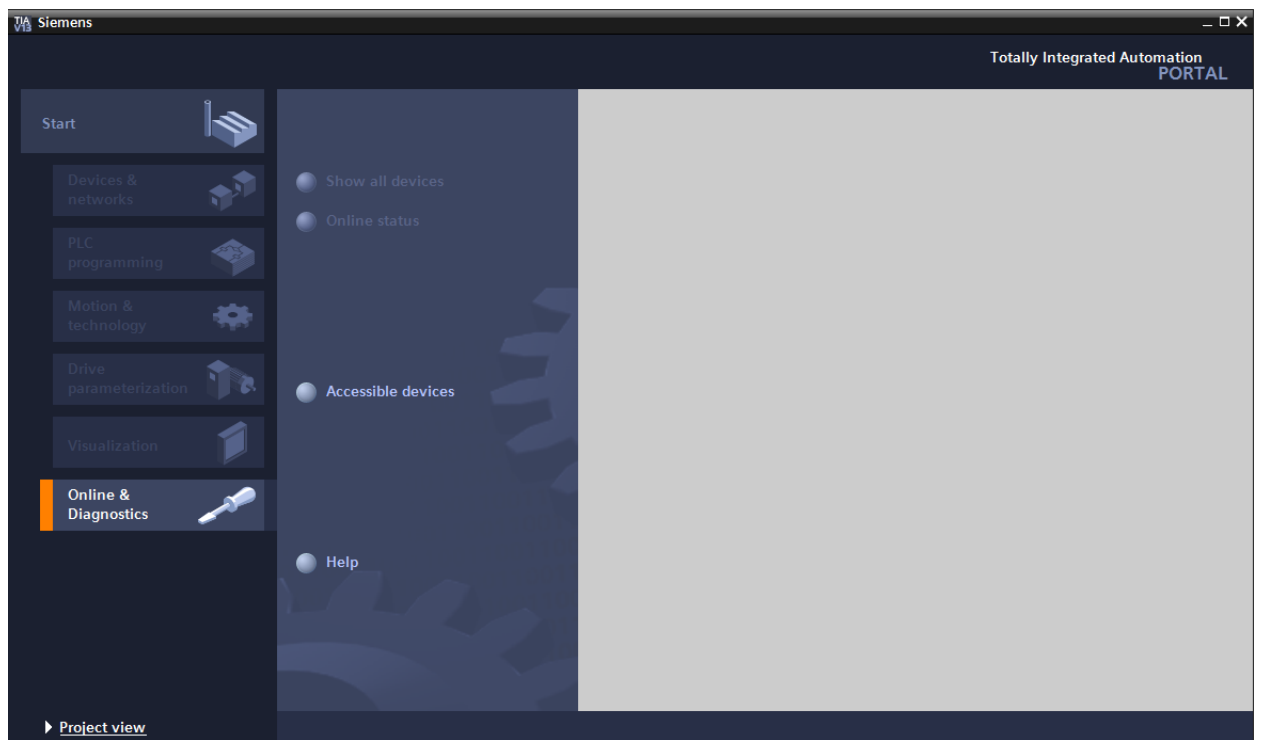
3.4.8 Setting the IP address in the CPU

The IP address of SIMATIC S7-1500 is set as follows.

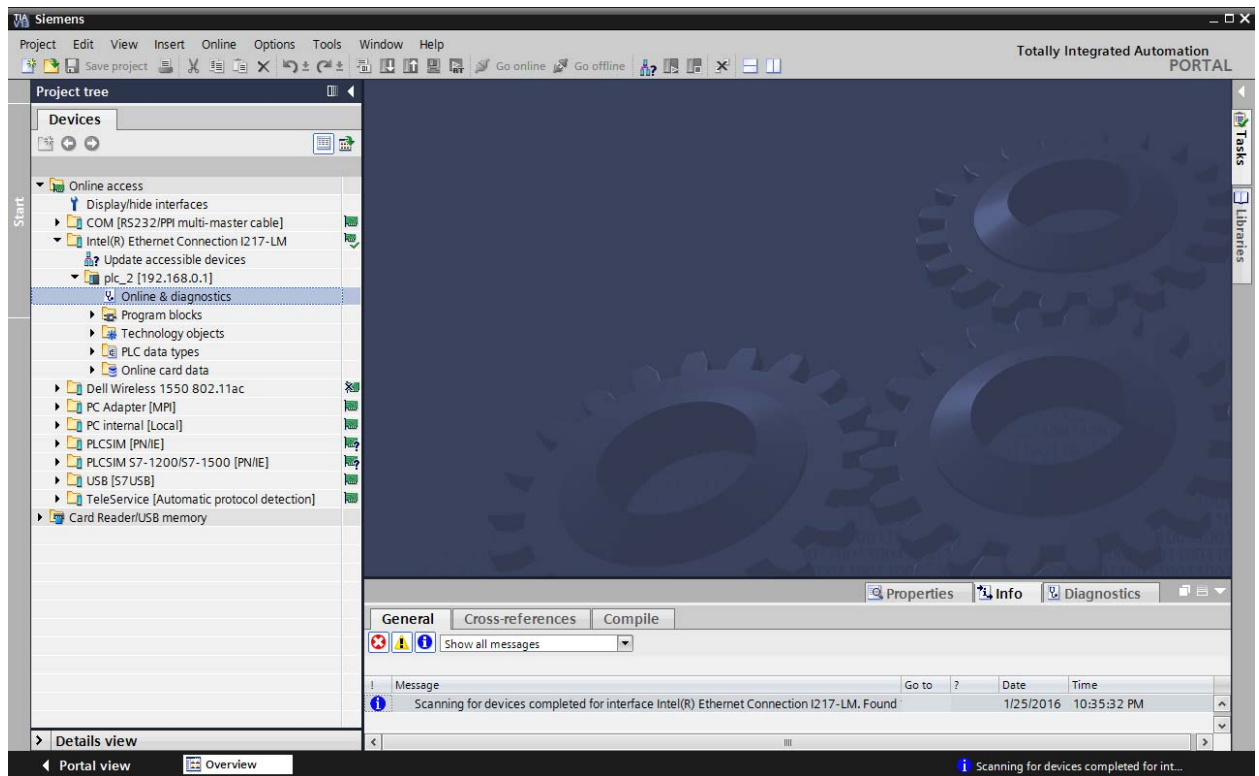
- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V13)



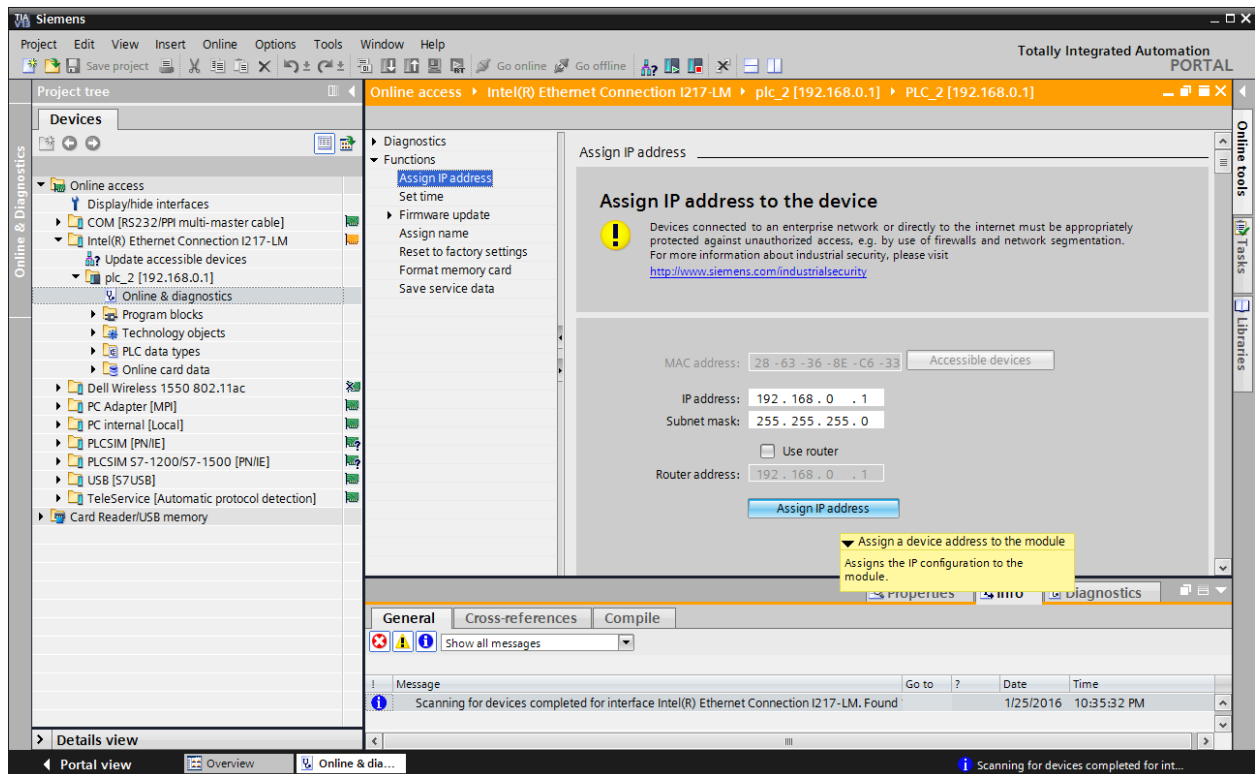
- Select → "Online & Diagnostics" and open the → "project view".



- In the project tree under → "Online access", select the network adapter that was set previously. If you click → "Update accessible devices" here, you will see the IP address (if previously set) or the MAC address (if IP address not yet assigned) of the connected SIMATIC S7-1500. Select → "Online & Diagnostics" here.

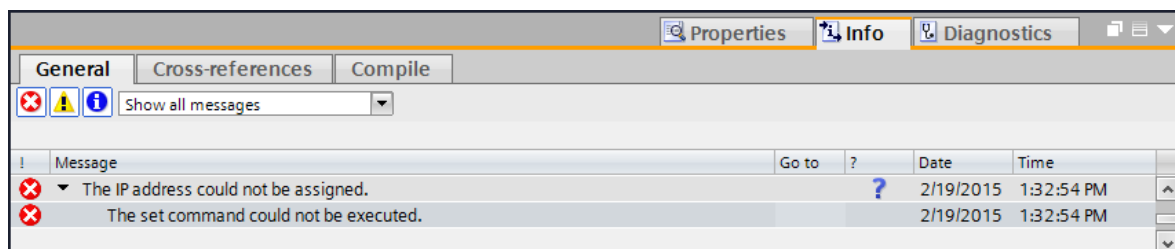


- Under → "Functions", you now find the → "Assign IP address" item. Enter the following IP address here (example): → IP address: 192.168.0.1 → Subnet mask 255.255.255.0. Next, click → "Assign IP address" and this new address will be assigned to your SIMATIC S7-1500.



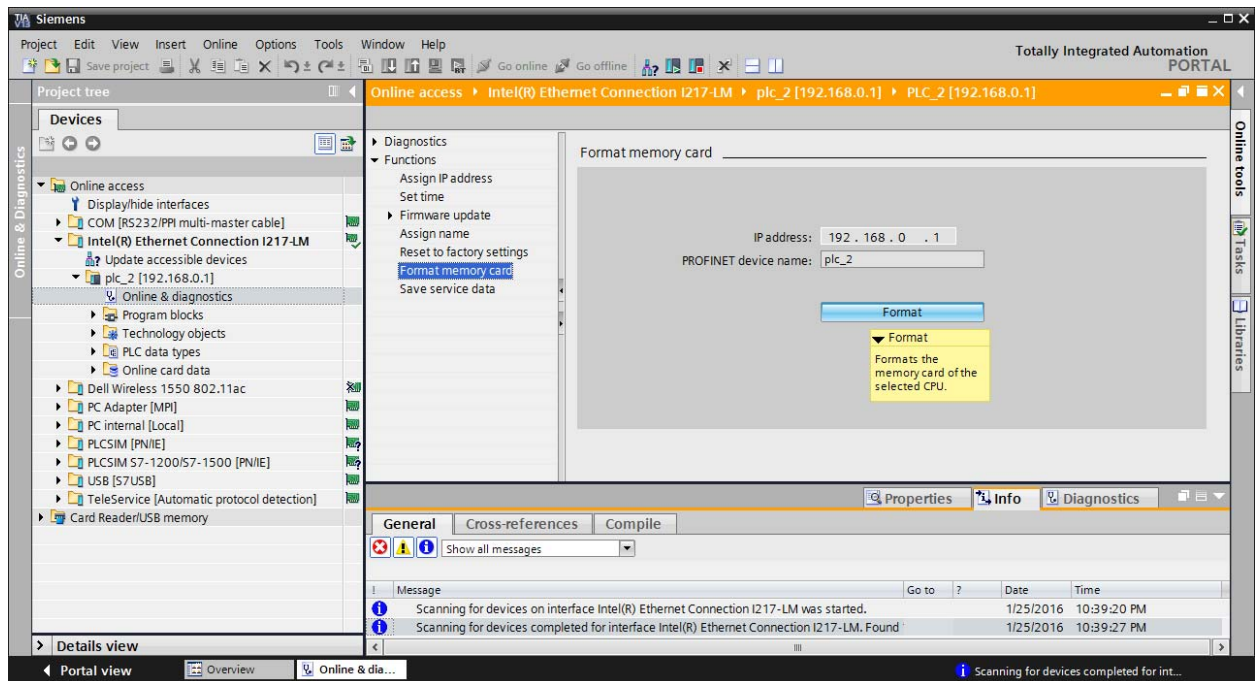
Note: The IP address of the SIMATIC S7-1500 can also be set via the display on the CPU, provided this is enabled in the hardware configuration.

- If the IP address was not successfully assigned, you will receive a message in the → "Info" window under → "General".

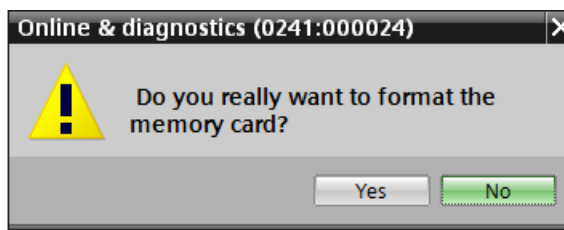


3.4.9 Formatting the memory card in the CPU

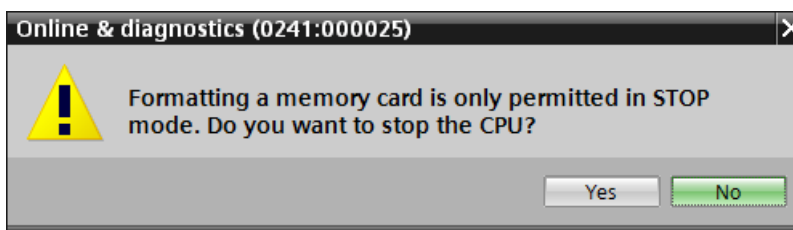
- If the IP address could not be assigned, the program data on the CPU must be deleted. This is accomplished in 2 steps: → "Format memory card" and → "Reset to factory settings".
- First, select the → "Format memory card" function and press the → "Format" button.



- Confirm the prompt asking if you really want to format the memory card with → "Yes".

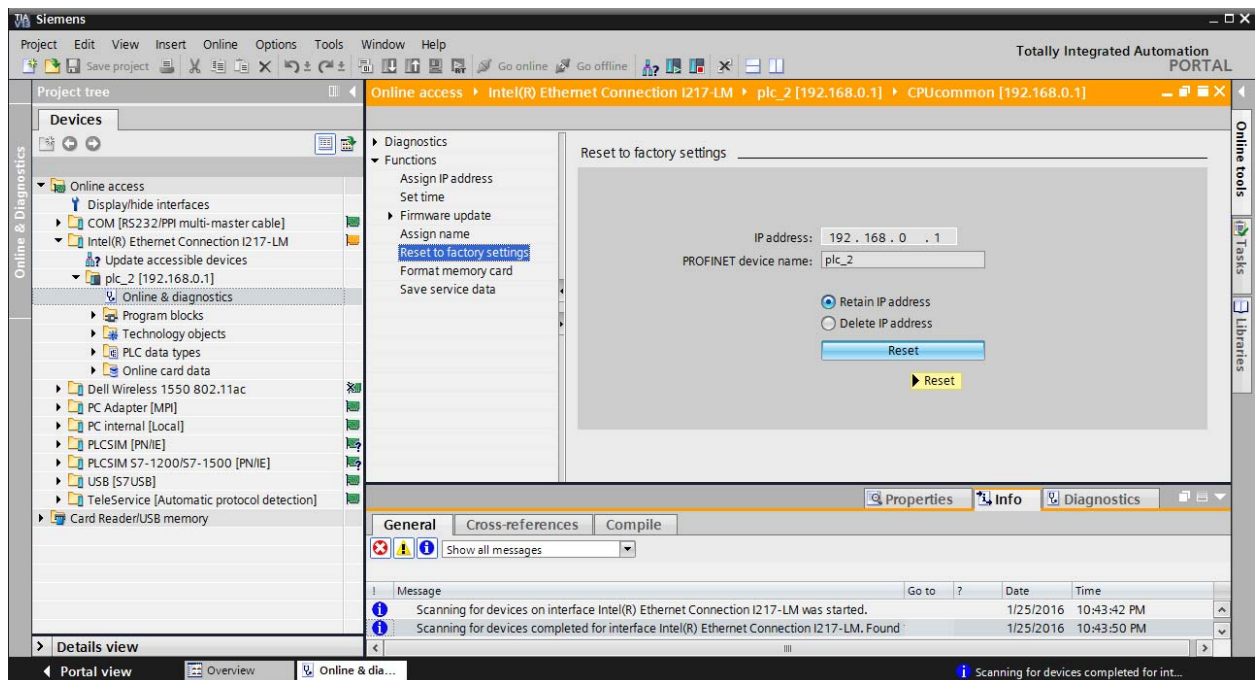


- If necessary, stop the CPU. (→ "Yes")

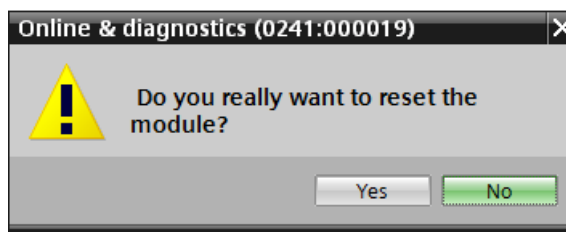


3.4.10 Resetting the CPU to factory settings

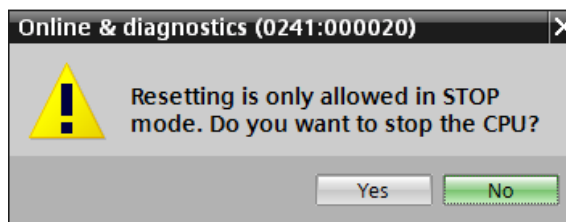
- Before you can reset the CPU, you must wait until the formatting in the CPU has finished.
- Then you must select → "Update accessible devices" and → "Online & diagnostics" of your CPU again. To reset the controller, select the → "Reset to factory settings" function and click → "Reset".



- Confirm the prompt asking if you really want to reset the module with → "Yes".



- If necessary, stop the CPU. (→ "Yes")



4 Task

Create a project and configure the following modules of your hardware, which correspond to one part of the Trainer Package **SIMATIC CPU 1512C PN with Software, PM 1507 and CP 1542-5 (PROFIBUS)**.

- 1X SIMATIC PM 1507 24 V/8 A STABILIZED POWER SUPPLY INPUT: 120/230 V AC OUTPUT: 24 V DC/8 A (ORDER NUMBER: 6EP1333-4BA00)
- SIMATIC S7-1500 COMPACT CPU, CPU 1512C-1 PN, CENTRAL PROCESSING UNIT WITH WORKING MEMORY 250 KB FOR PROGRAM AND 1 MB FOR DATA, 32 DIGITAL INPUTS, 32 DIGITAL OUTPUTS, 5 ANALOG INPUTS, 2 ANALOG OUTPUTS, 6 HIGH SPEED COUNTERS, 1. INTERFACE: PROFINET IRT WITH 2 PORT SWITCH, 48 NS BIT PERFORMANCE, INCL. FRONT CONNECTOR, SIMATIC MEMORY CARD REQUIRED (ORDER NUMBER: 6ES7 512-1CK00-0AB0)

5 Planning

Because this is a new system, a new project must be created.

The hardware is already specified for this project with the trainer package SIMATIC CPU 1512C PN with Software, PM 1507 and CP 1542-5 (PROFIBUS) Therefore, no selection needs to be made. Instead, the listed modules of the Trainer Package are simply inserted in the project.

To ensure that the correct modules are inserted, re-check the order numbers from the task directly on the installed devices. Normally, the CPU is inserted first. Other signal modules for the compact CPU are not needed here. Then add the power supply. See Table 1.

The Ethernet interface must be set for the configuration of the CPU. In addition, the address areas of the digital and analog inputs and outputs must be adapted.

Module	Order number	Slot	Address area
PM 190W 120/230VAC	6EP1333-4BA00	0	
CPU 1512C-1 PN	6ES7 512-1CK00-0AB0	1	DI 0...3 DQ 0...3 AI 64...73 AQ 64...67

Table 1: Overview of the planned configuration

As the final step, the hardware configuration must be compiled and downloaded. Any errors present can be detected during compilation and incorrect modules can be detected when the controller is started (*only possible when hardware is present and structured identically*). Now, back up the checked project.

6 Structured step-by-step instructions

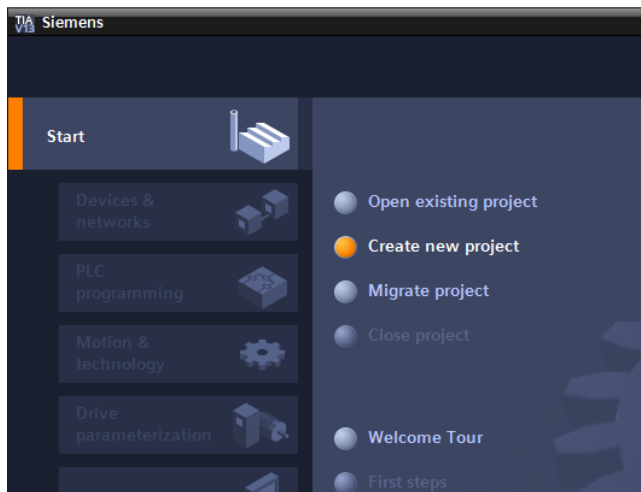
You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it is sufficient to focus on the numbered steps. Otherwise, simply follow the steps of the instructions illustrated below.

6.1 Create a new project

- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V13)



- In the portal view under the "Start" menu, select → "Create new project".

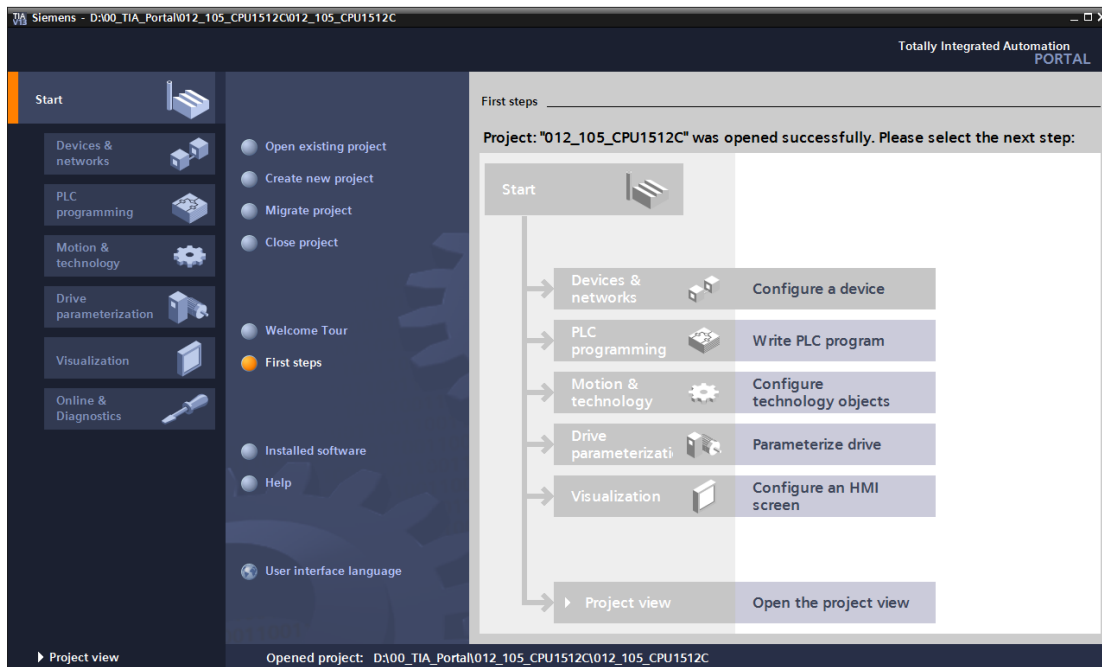


- Modify Project name, Path, Author and Comment as appropriate and click → "Create".

- The project will be created and opened and the menu "Start", "First steps" will open automatically.

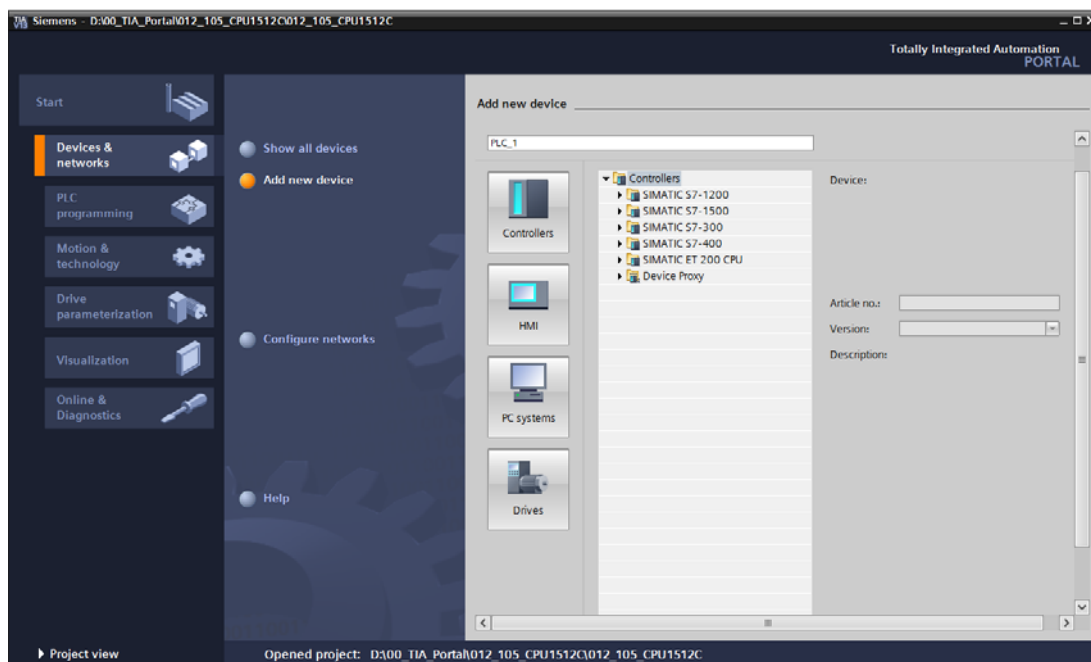
6.2 Inserting the CPU 1512C-1 PN

→ In the portal, select → "Start" → "First steps" → "Devices & networks" → "Configure a device".



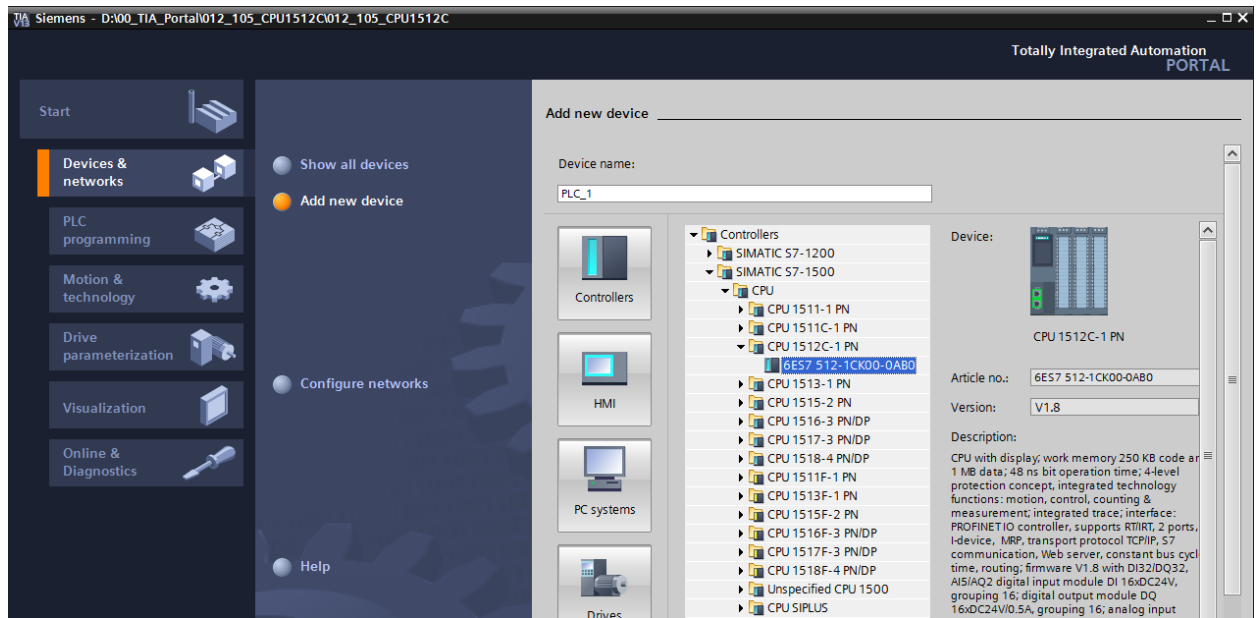
→ In the portal, "Devices & networks" opens the "Show all devices" menu.

→ Switch to the "Add new device" menu.



→ The specified model of the CPU will now be added as a new device.

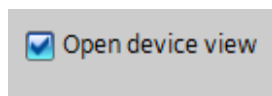
(Controller → SIMATIC S7-1500 → CPU → CPU 1512C-1 PN → 6ES7512-1CK00-0AB0 → V1.8)



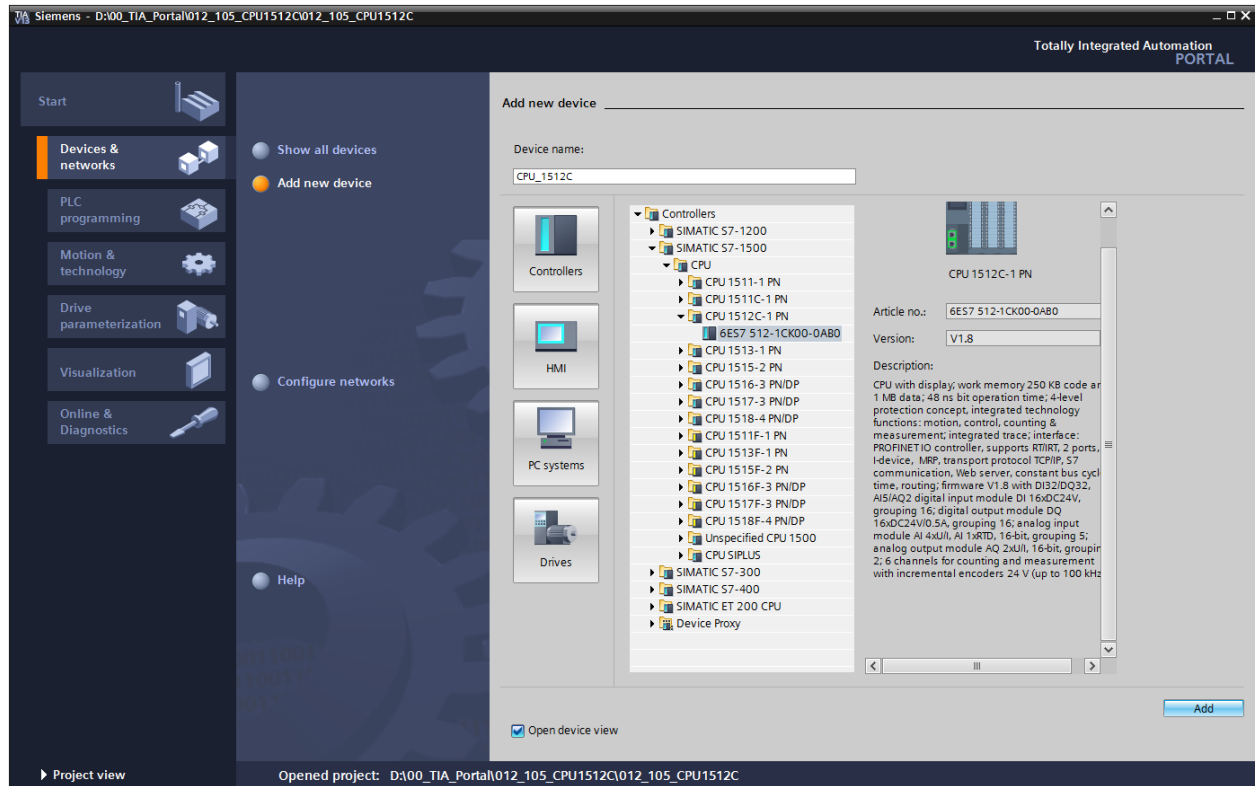
→ Assign a device name. (device name → "CPU_1512C")



→ Select "Open device view".



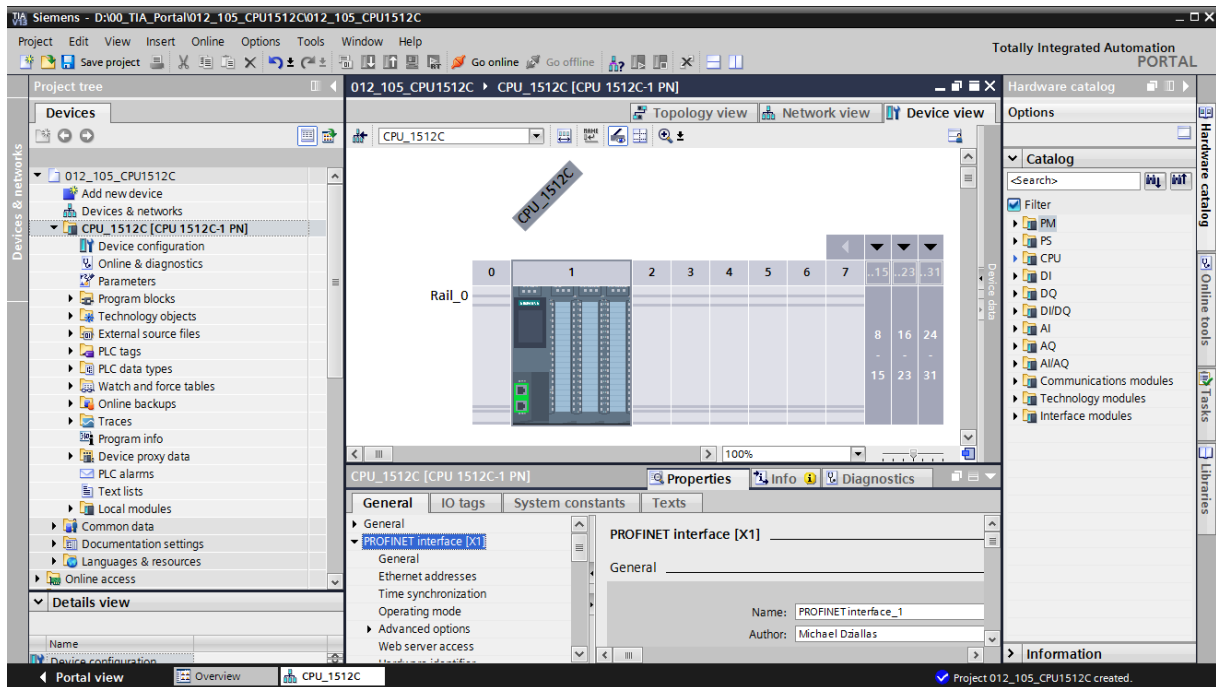
→ Click "Add".



Note: The desired CPU may have multiple versions that differ in functionality (work memory, integrated memory, technology functions, etc.). In this case, you should ensure that the selected CPU corresponds to the existing hardware.

Note: Different firmware versions are often offered for the hardware. In this case, it is recommended to use the latest firmware (selected by default) and to upgrade the CPU, if necessary.

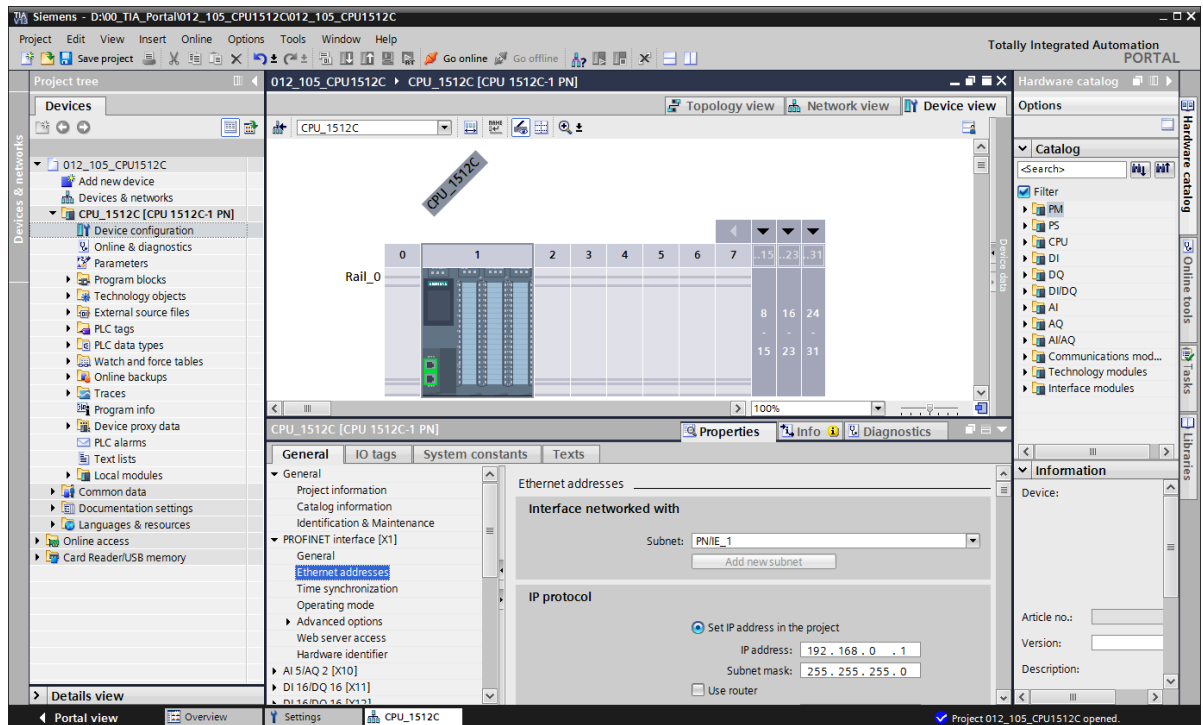
→ The TIA Portal now changes automatically to the project view and displays the selected CPU in the device configuration in slot 1 of a rail.



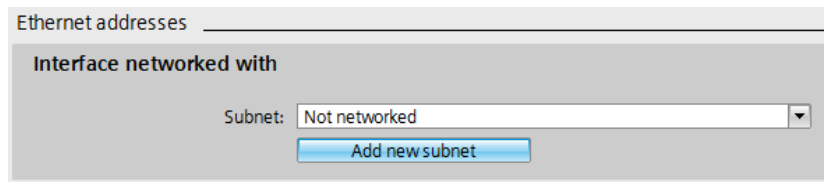
Note: You can now configure the CPU there according to your specifications. Possible settings include the PROFINET interface, startup characteristics, cycle, communication load and many others.

6.3 Configure Ethernet interface of the CPU 1512C-1 PN

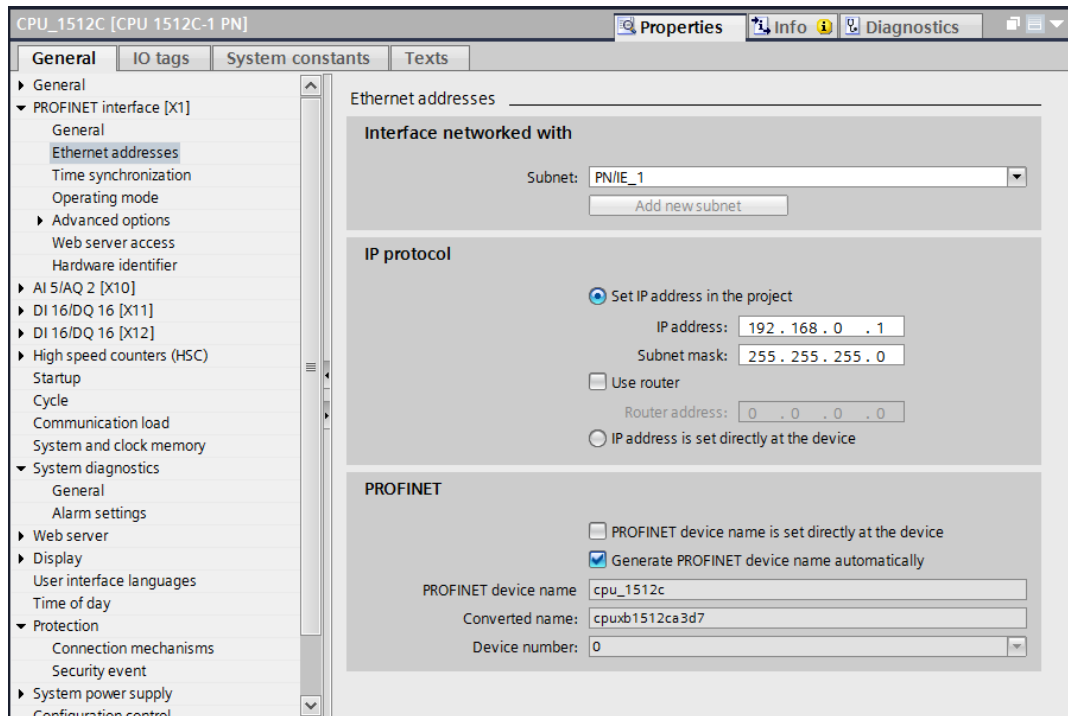
- Select the CPU with a double-click
- Under → "Properties", open the → "PROFINET interface [X1]" menu and select the → "Ethernet addresses" entry there.



- Under "Interface networked with", only the "Not networked" entry is available.
- Add an Ethernet subnet with the → "Add new subnet" button.

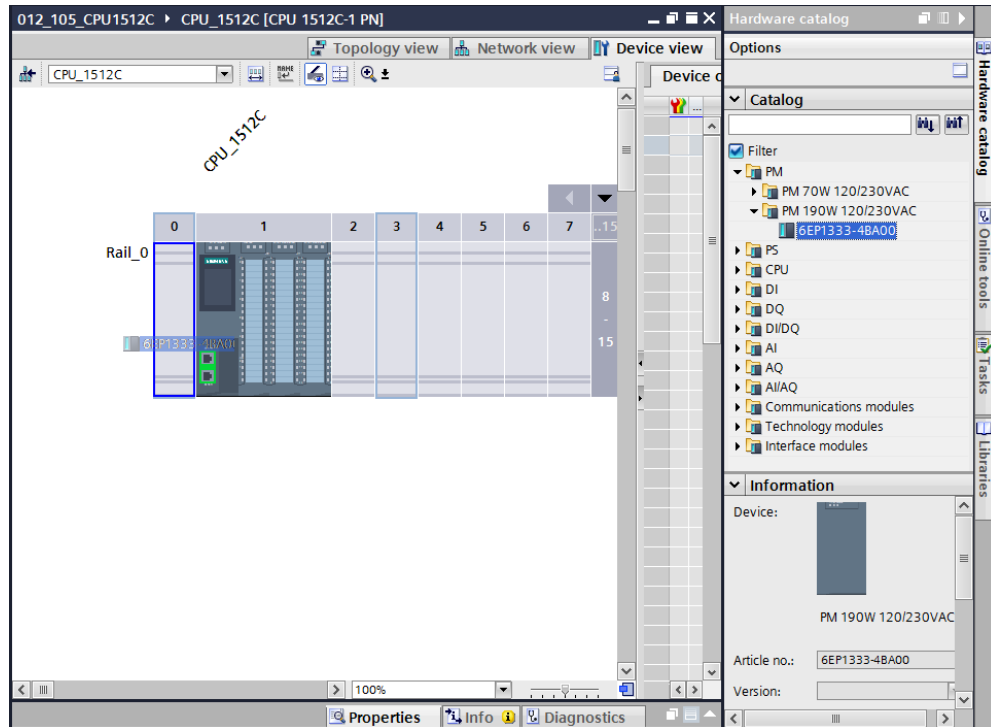



→ Keep the preassigned "IP address" and "Subnet mask".

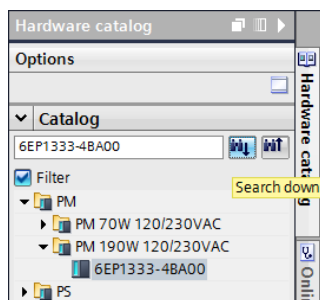


6.4 Insert power module PM 190W 120/230VAC

- Find the correct module in the hardware catalog and insert the power module in slot 0.
 (→ Hardware Catalog → PM → PM 190W 120/230VAC (order number 6EP1333-4BA00) → Slot 0)



Note: To select the power module, you can simply enter the order number in the Search field and then click the "Search down"  icon. The hardware catalog will open at the correct position.



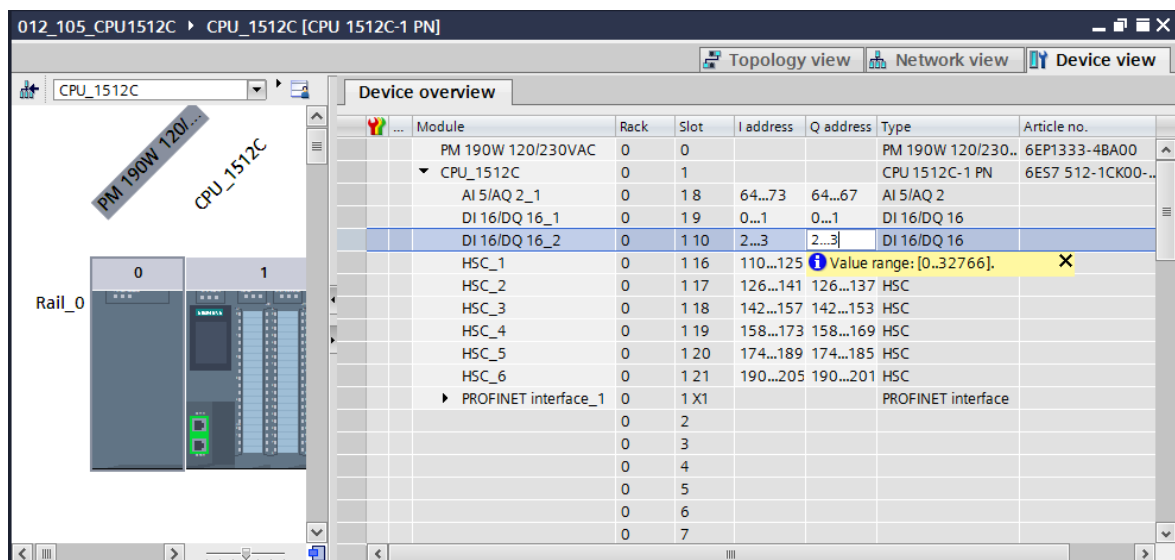
Notes: When you double-click a module in the hardware catalog, you insert it at the next available compatible slot.

If a module such as the power module is planned only for one slot, it is not possible to place it at another position even in the device configuration.

6.5 Configure the address areas for the digital and analog inputs and outputs

The address areas for the signals used in the process are specified by the plant designer to correspond to the plant documentation. See Section 5 Planning.

- In the "Device overview" section, ensure that the address areas of the high-speed counters (HS) not used here are above 100.
(→ Device overview → HSC_1 to HSC_6 → I address → 110... 205 → Q address → 110...201)
- Set the analog inputs and outputs to the correct address areas starting at 64....
(→ Device overview → AI5/AQ2 → Slot 1 8 → I address → 64...73 → Q address → 64...67)
- Set the digital inputs and outputs to the correct address areas starting at 0.... (→ Device overview → DI16/DQ16 → Slot 1 9 → I address → 0...1 → Q address → 0...1
→ DI16/DQ16 → Slot 1 10 → I address → 2...3 → Q address → 2...3)



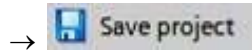
Note: To show and hide the Device overview, you must click the small arrow next to "Device data" on the right side of the hardware configuration.



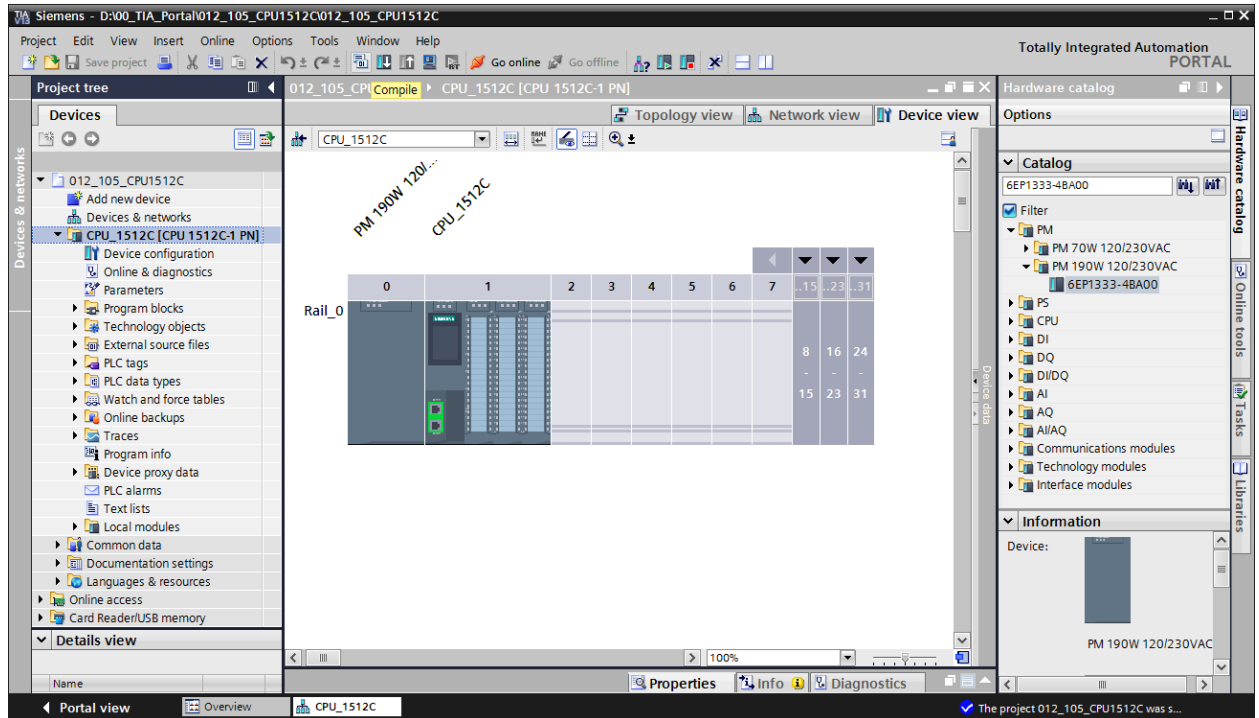
Note: To set the address areas for the digital and analog inputs and outputs as desired, the address areas for the high-speed counters (HSC) must first be moved to higher areas.

6.6 Save and compile the hardware configuration

→ Before you compile the configuration, you should save your project by clicking the

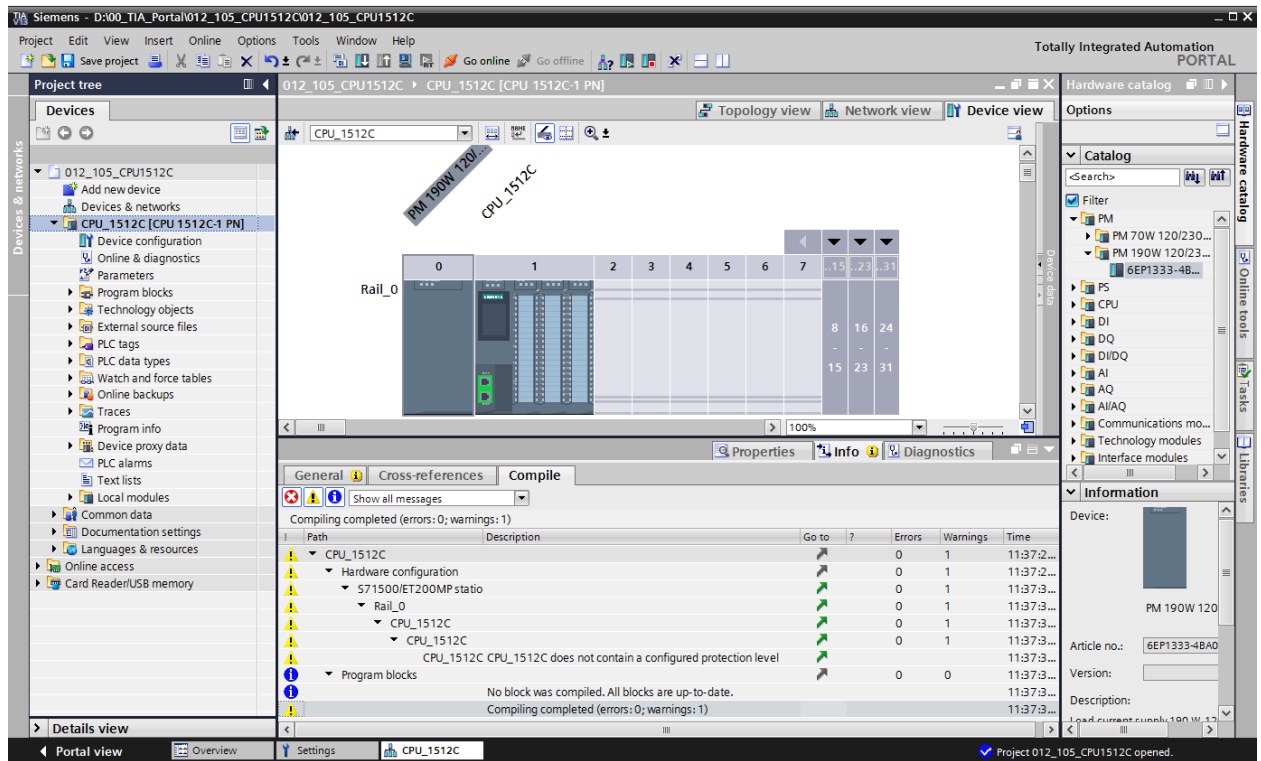


button. To compile your CPU with the device configuration, first select the → "CPU_1512C [CPU1512C-1 PN]" folder and click the → "Compile" icon.




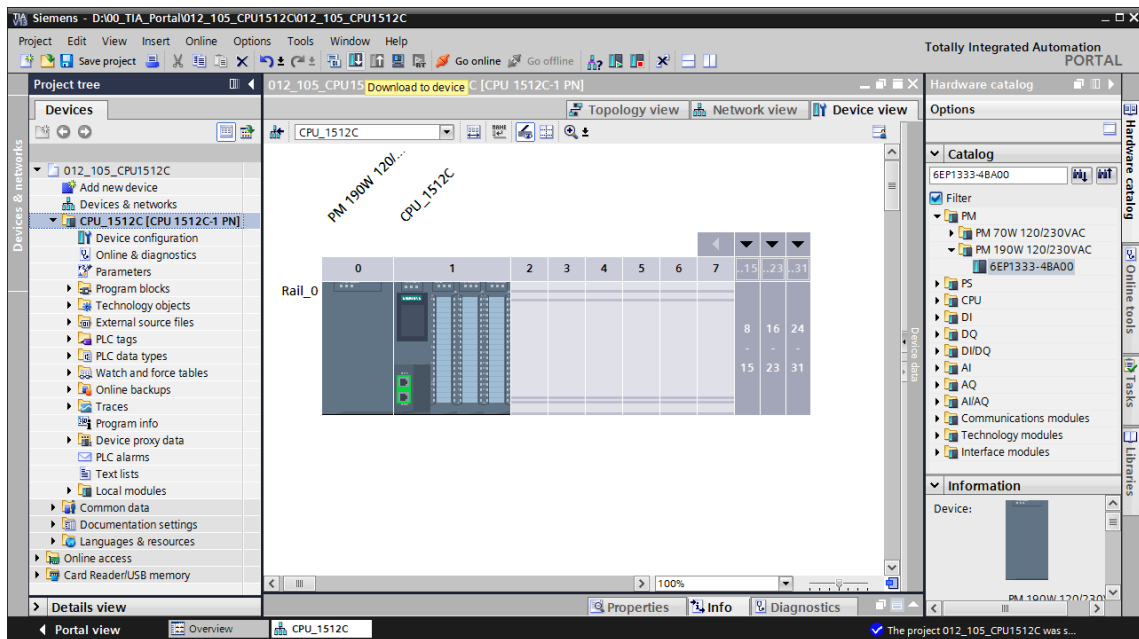
Note: "Save project" should be used repeatedly when working on a project since this does not happen automatically. A prompt to save the project only occurs when the TIA Portal is closed.

→ If the project was compiled without errors, you see the following screen.

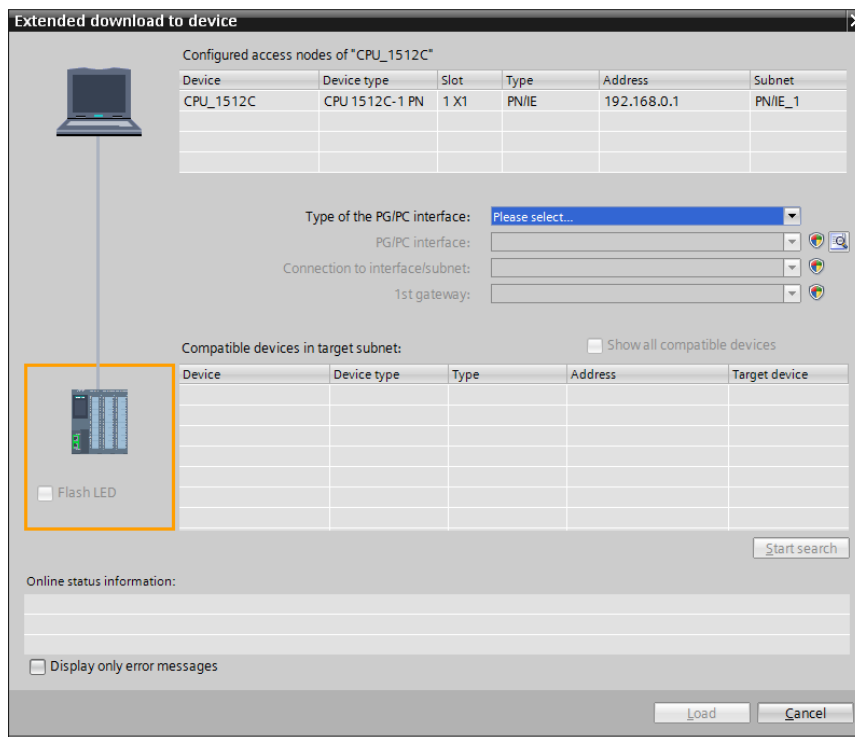


6.7 Download the hardware configuration to the device

→ To download your entire CPU, select the → "CPU_1512C [CPU1512C-1 PN]" folder and click the  → "Download to device" icon.

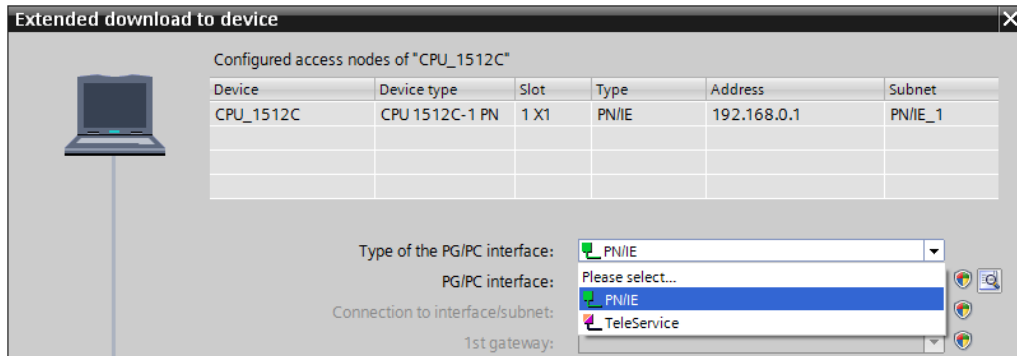


→ The manager for configuring the connection properties (extended download) opens.

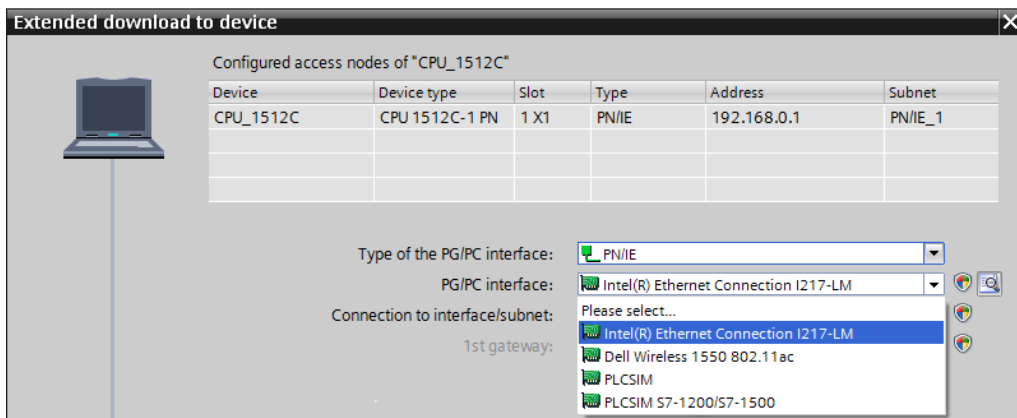


→ First, the interface must be correctly selected. This happens in three steps.

→ Type of the PG/PC interface → PN/IE

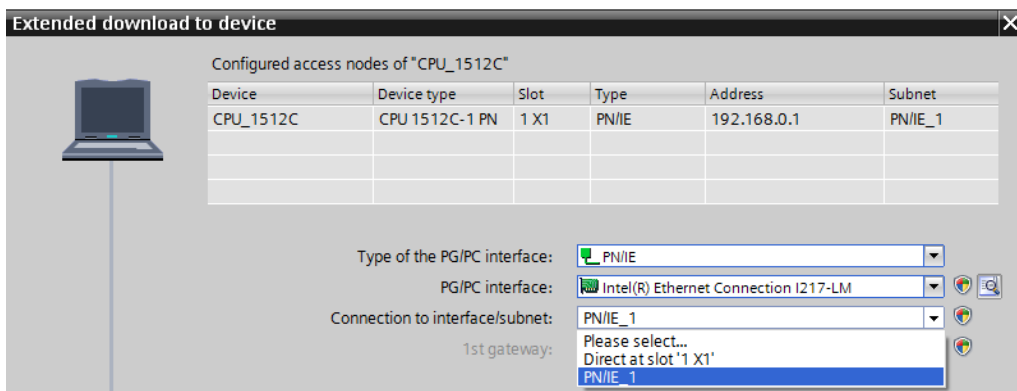


→ PG/PC interface → here, for example: Intel(R) Ethernet Connection I217-LM

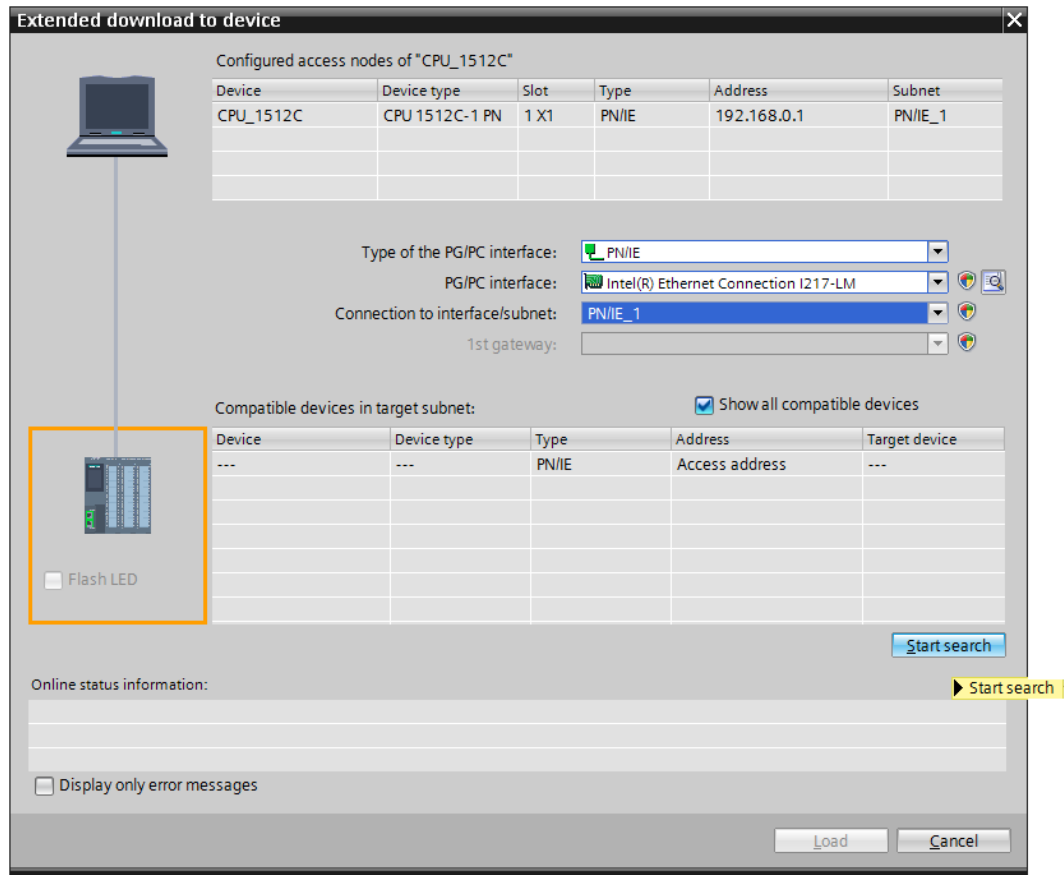


Note: The PG/PC interface used here depends on the ETHERNET interface adapter installed in the programming device

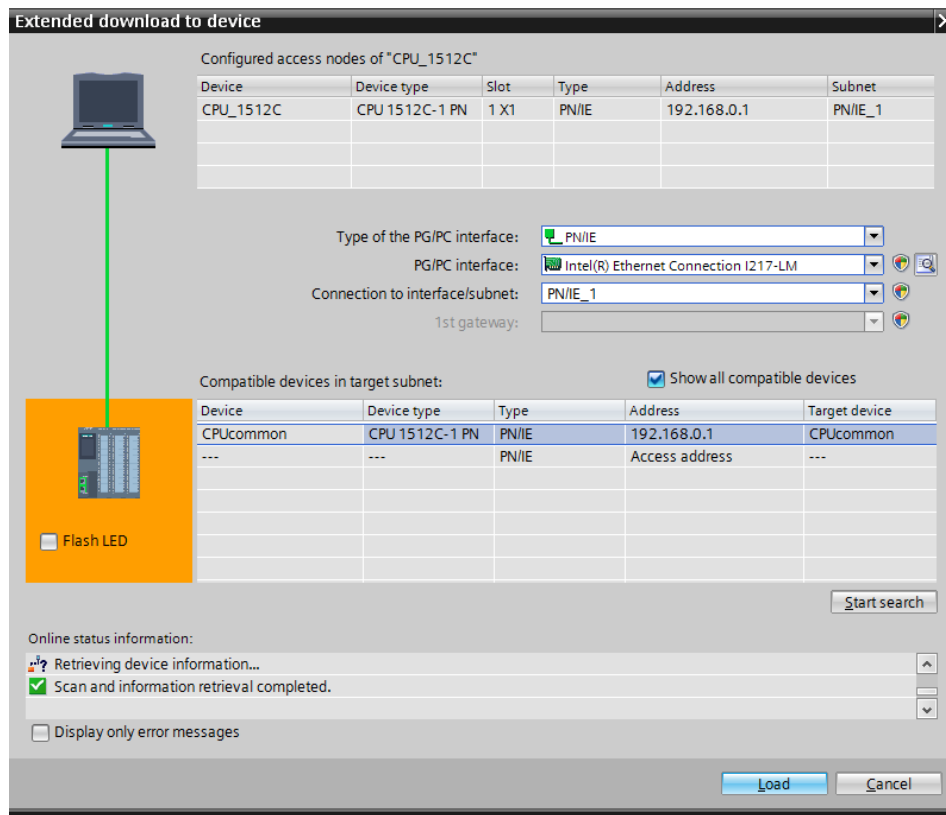
→ Connection to interface/subnet → "PN/IE_1"



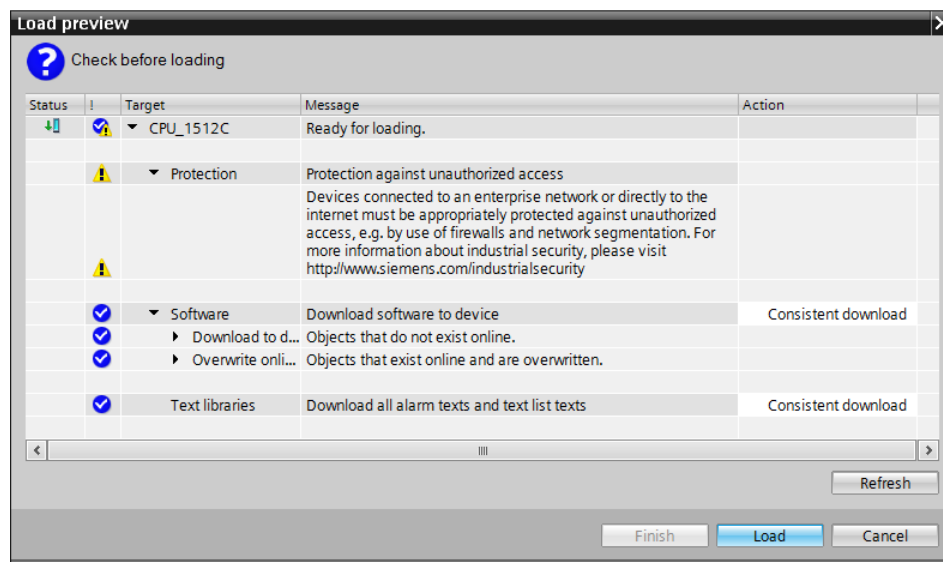
→ The → "Show all compatible devices" check box must then be selected. The search for devices in the network is started by clicking the → **Start search** button.



- If your CPU is shown in the "Compatible devices in target subnet" list, it must be selected and the download started. (→ CPU 1512C-1 PN → "Load")

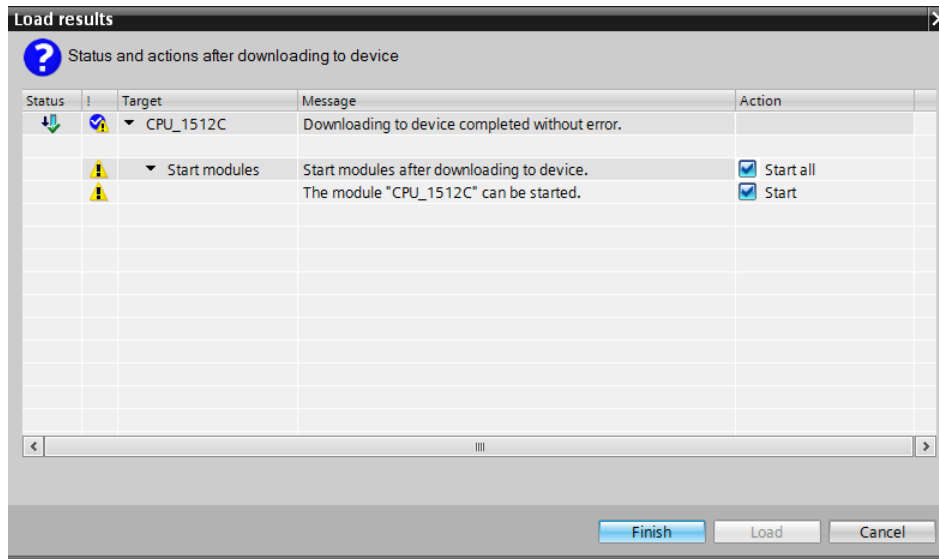


- You first obtain a preview. Confirm the prompt → "Overwrite all" and continue with → "Load".

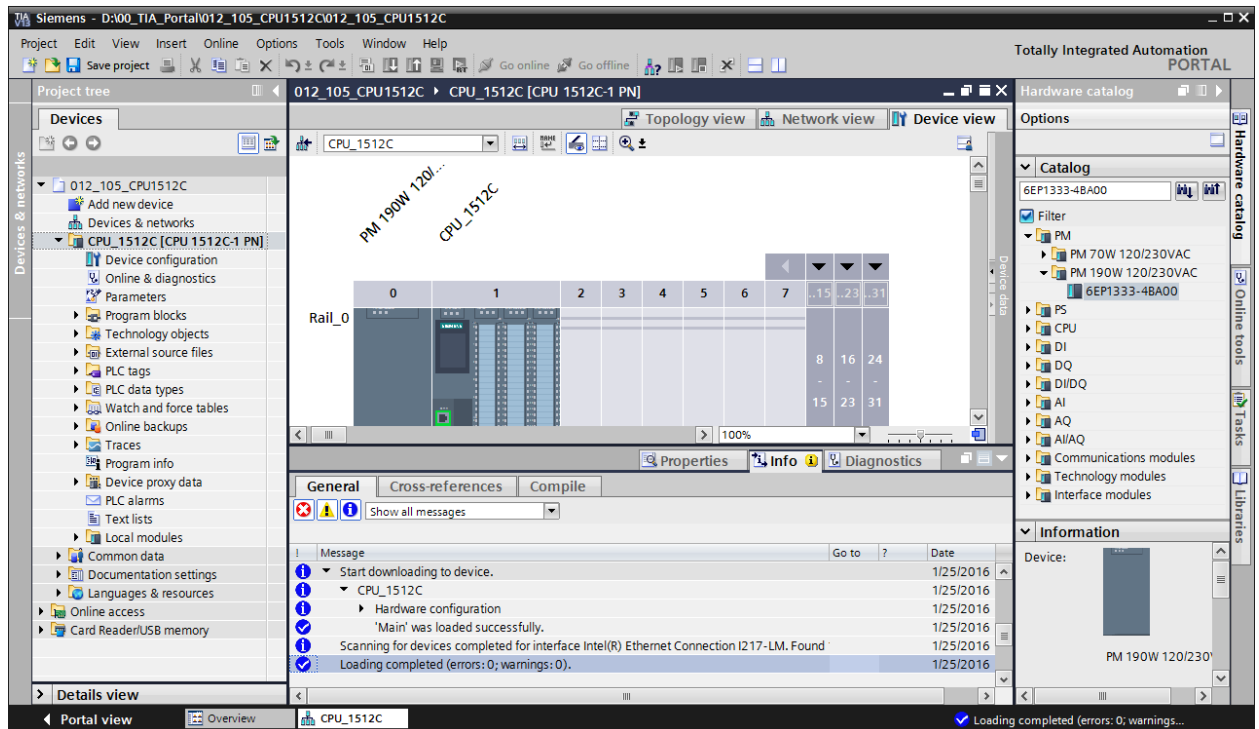


Note: The ☒ symbol should be visible in every line of the "Load preview". You can find additional information in the "Message" column.


→ The → "Start all" option will be selected next before the download operation can be completed with → "Finish".

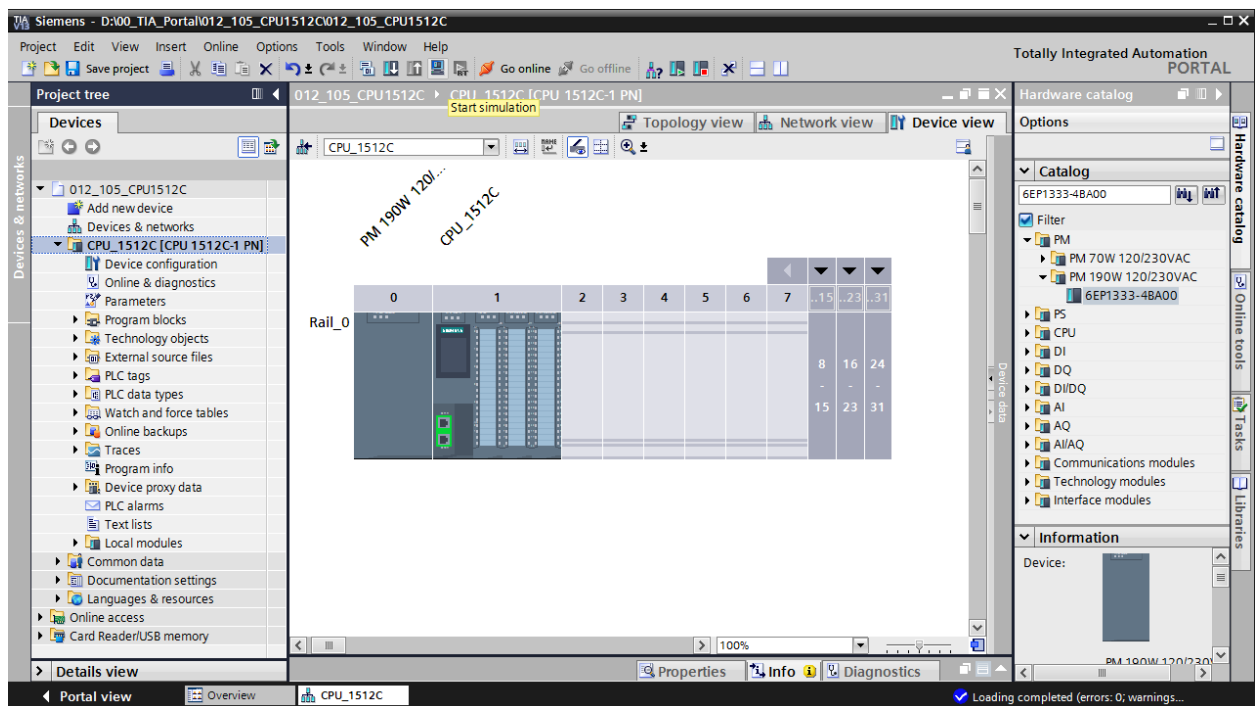


→ After a successful download, the project view will open again automatically. A loading report appears in the information field under "General". This can be helpful when troubleshooting an unsuccessful download.

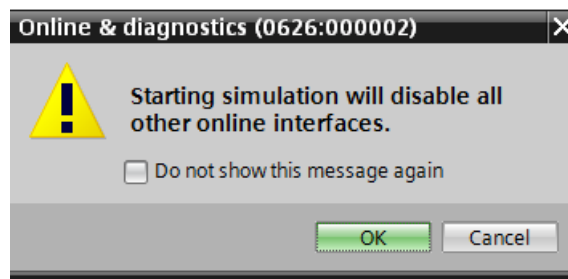


6.8 Download the hardware configuration to the PLCSIM simulation (optional)

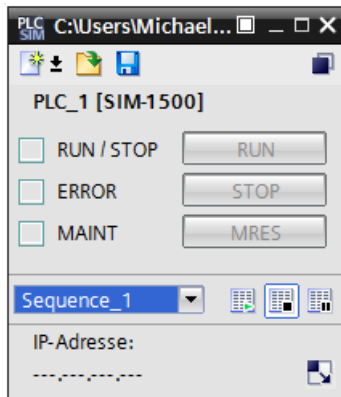
- If no hardware is present, the hardware configuration can **alternatively** be downloaded to a PLC simulation (S7-PLCSIM).
- To do so, you must first start the simulation by selecting the → "CPU_1512C [CPU1512C-1 PN]" folder and clicking the  → "Start simulation" icon.



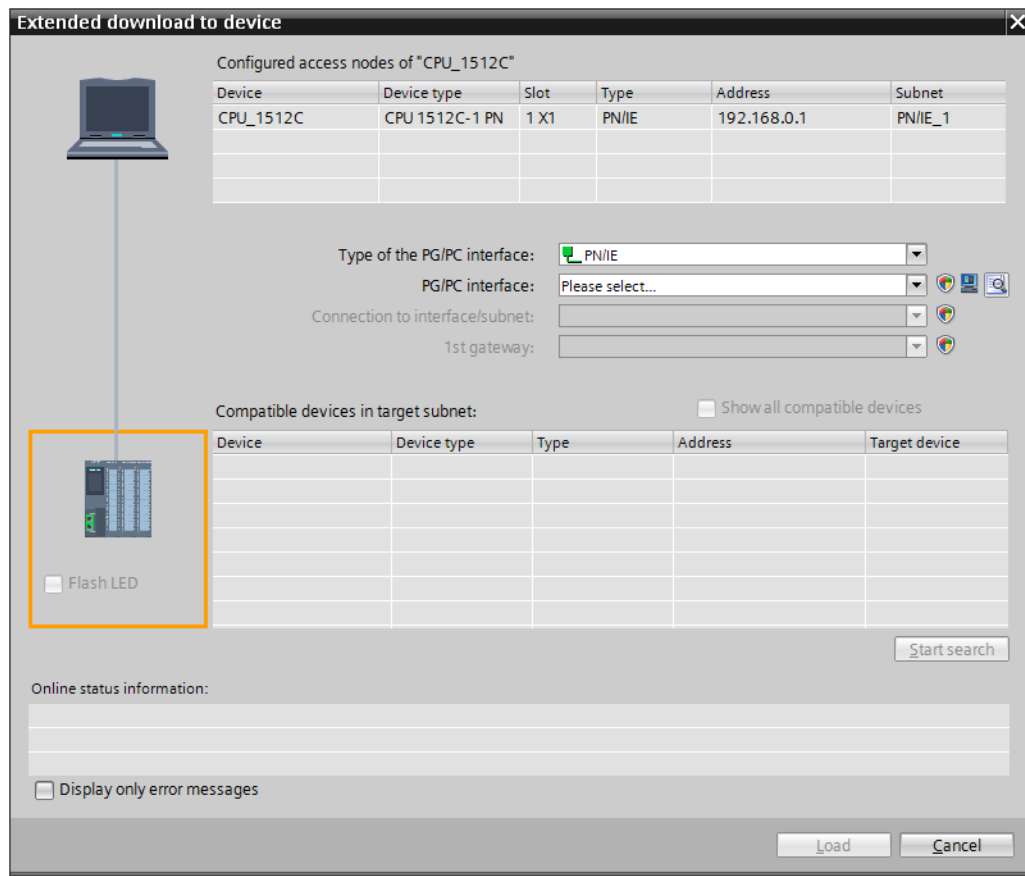
- The prompt that all other online interfaces will be disabled is confirmed with → "OK".



→ The "S7 PLCSIM" software is started in a separate window in the compact view.

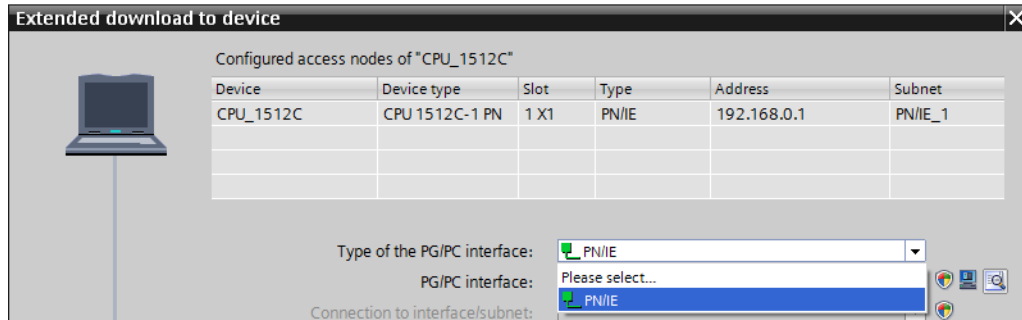


→ The manager for configuring the connection properties (extended download) opens shortly thereafter.

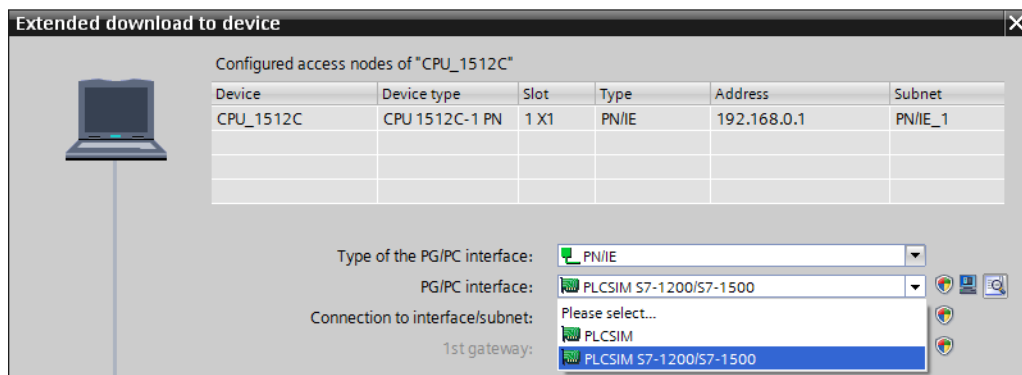


→ First, the interface must be correctly selected. This happens in three steps.

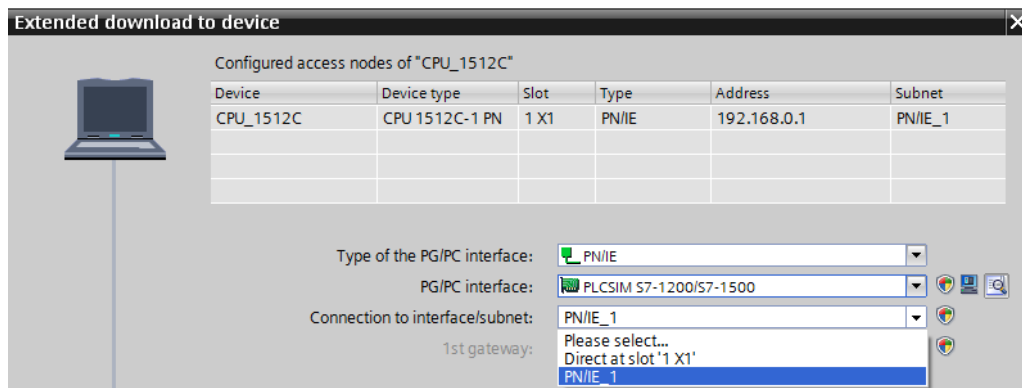
→ Type of the PG/PC interface → PN/IE



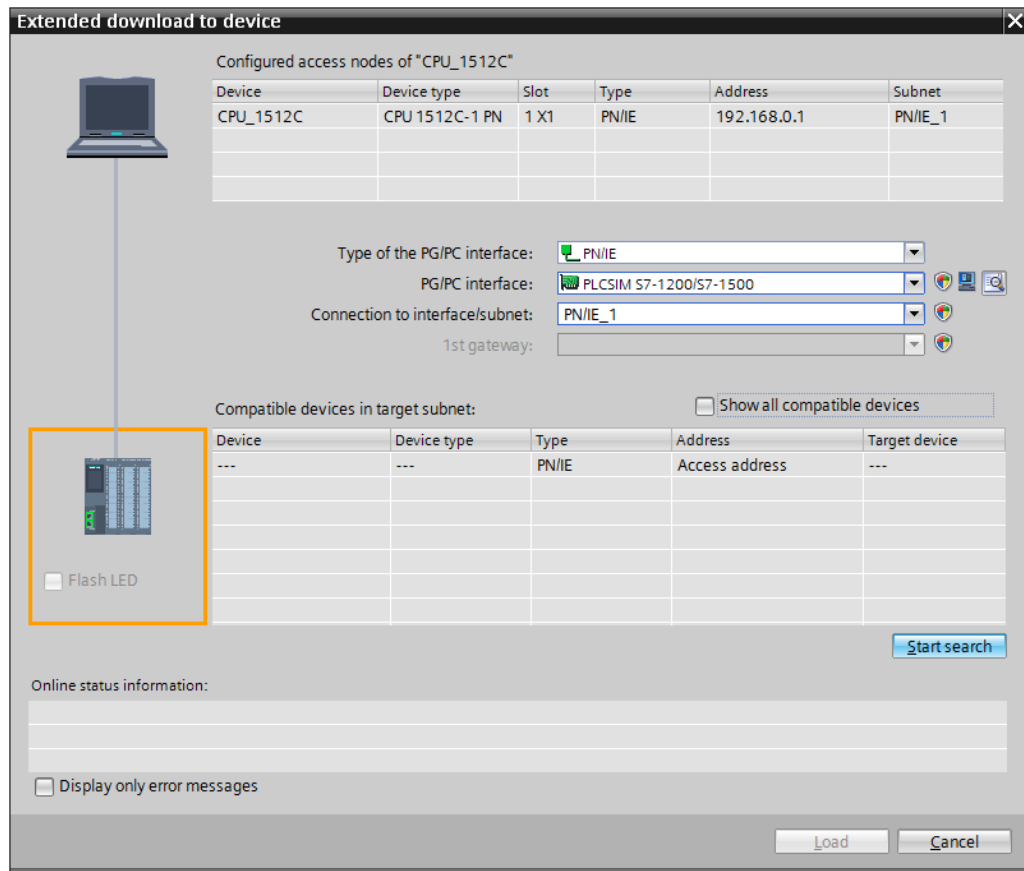
→ PG/PC interface → PLCSIM S7-1200/S7-1500



→ Connection to interface/subnet → "PN/IE_1"

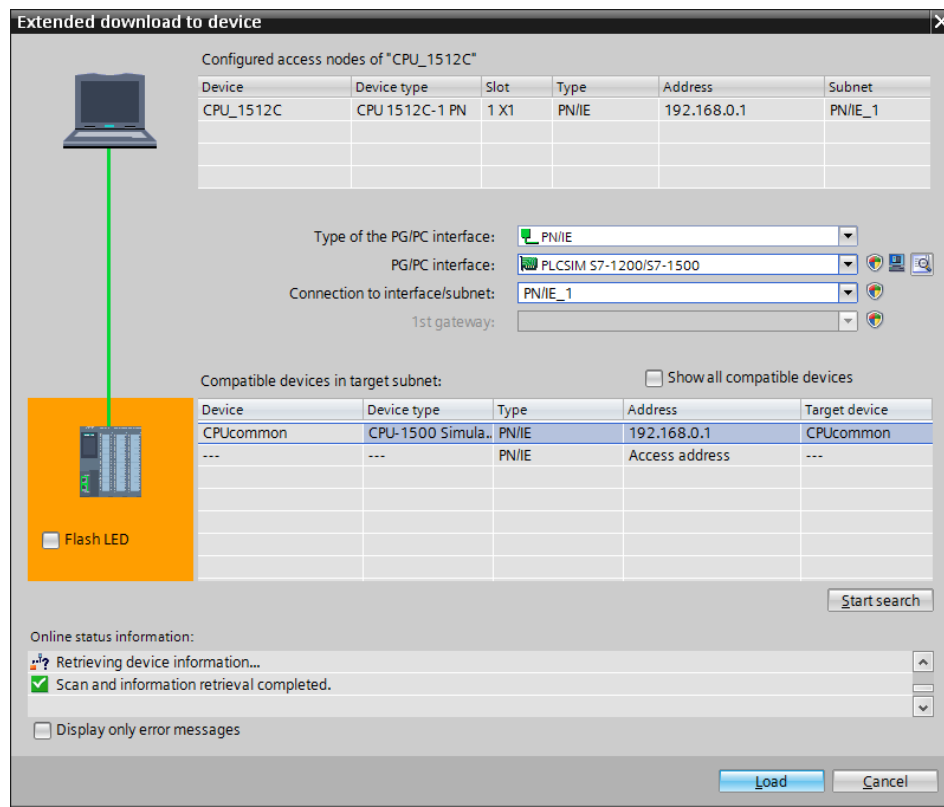


→ The → "Show all compatible devices" check box must then be selected. The search for devices in the network is started by clicking the → **Start search** button.

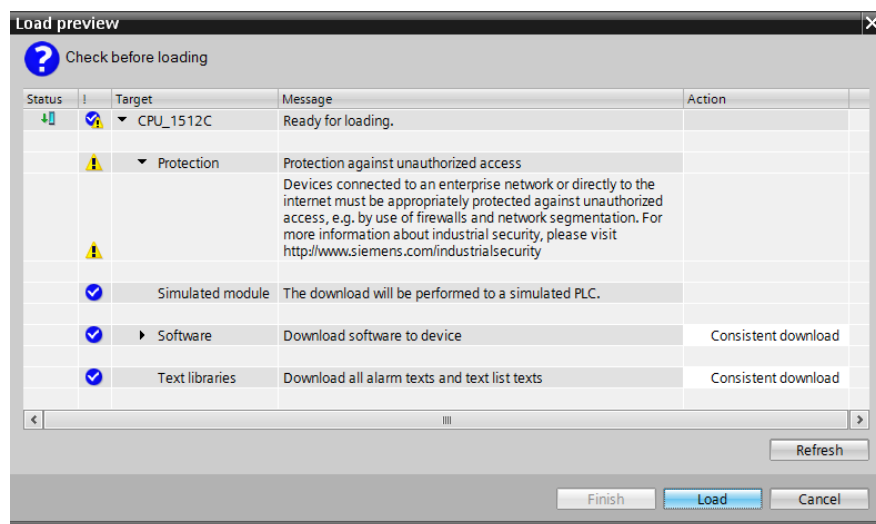


→ If the simulation is shown in the "Compatible devices in target subnet" list, it must be selected before the download can be started.

(→ "CPU-1500 Simulation" → "Load")

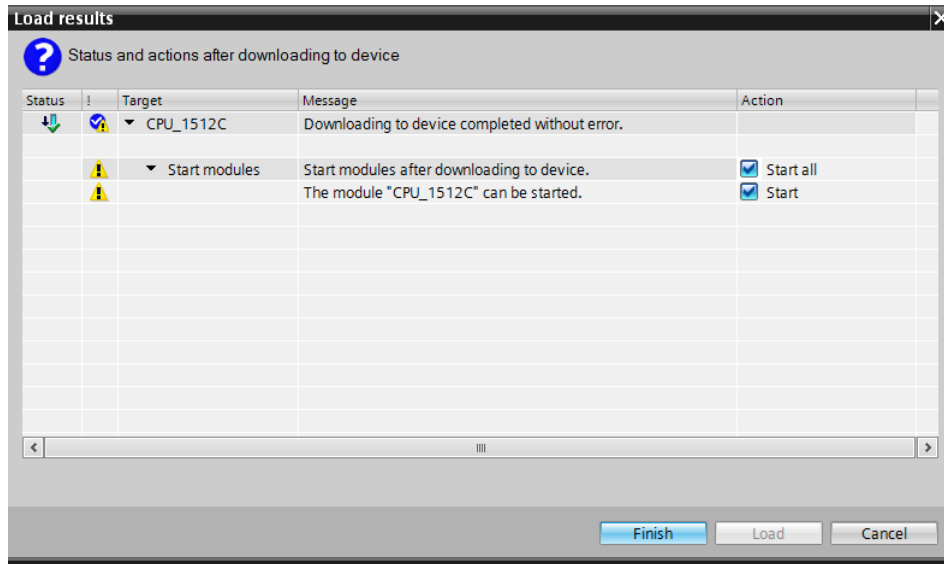


→ You first obtain a preview. Confirm the prompt → "Overwrite all" and continue with → "Load".

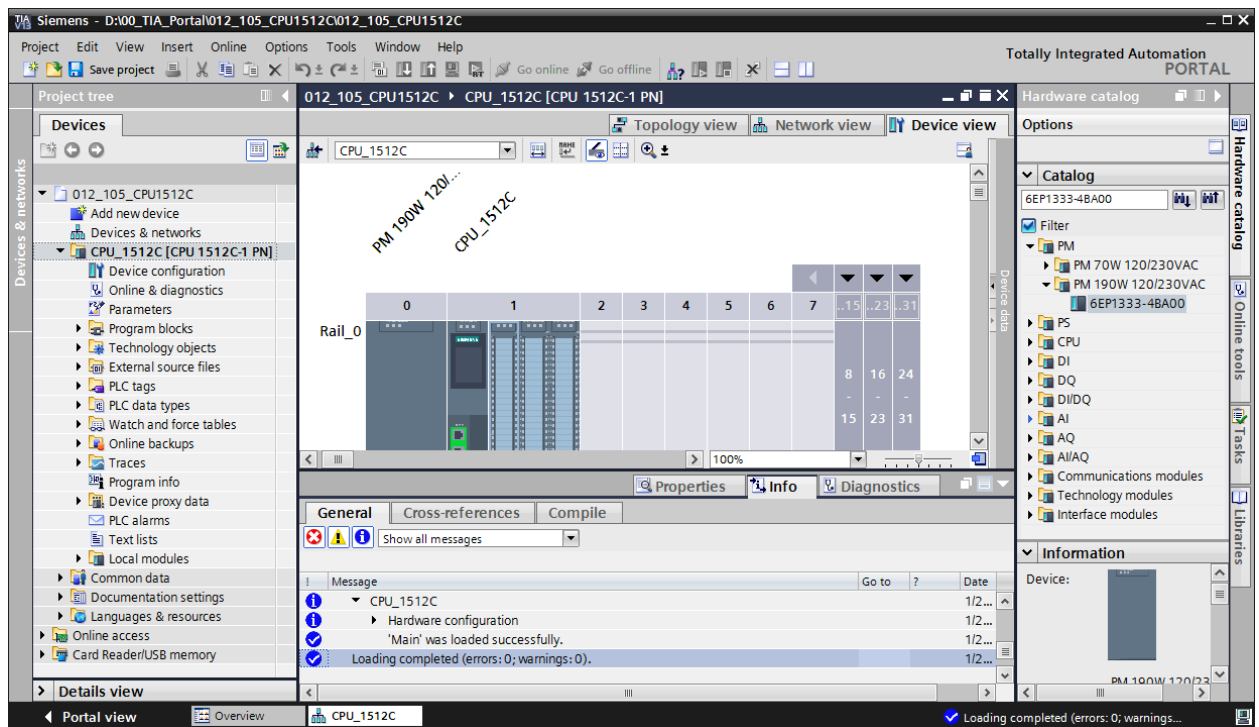



Note: The symbol should be visible in every line of the "Load preview". You can find additional information in the "Message" column.

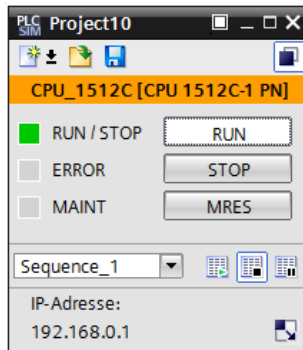
→ The → "Start all" option will be selected next before the download operation can be completed with → "Finish".




→ After a successful download, the project view will open again automatically. A loading report appears in the information field under "General". This can be helpful when troubleshooting an unsuccessful download.

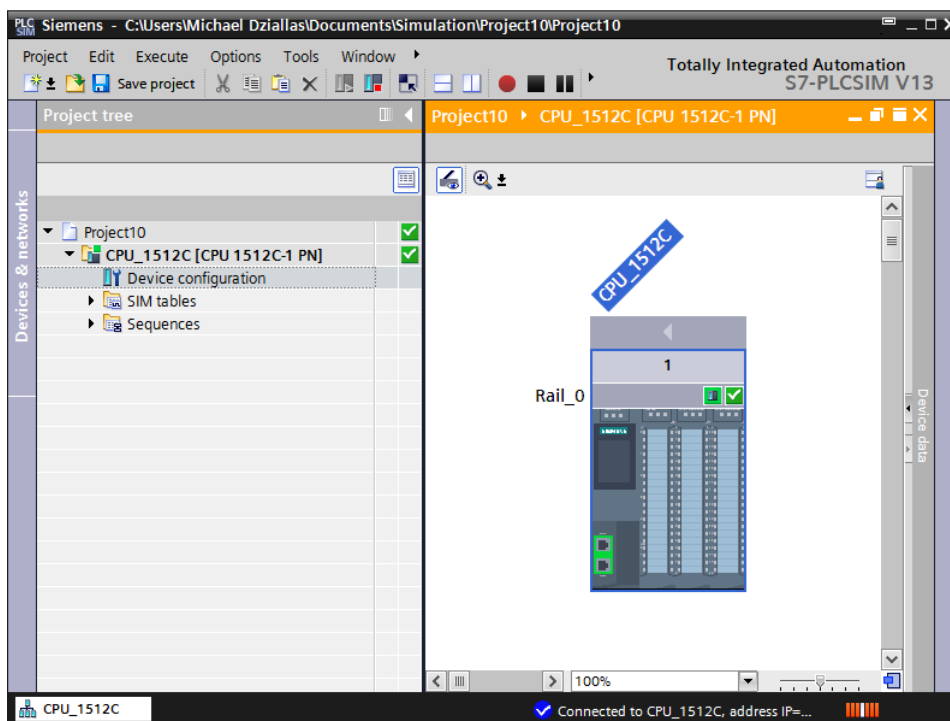


- The compact view of the PLCSIM simulation has the following appearance. You can switch back to the project view by clicking the →  icon.



The PLCSIM simulation has the following appearance in the project view.

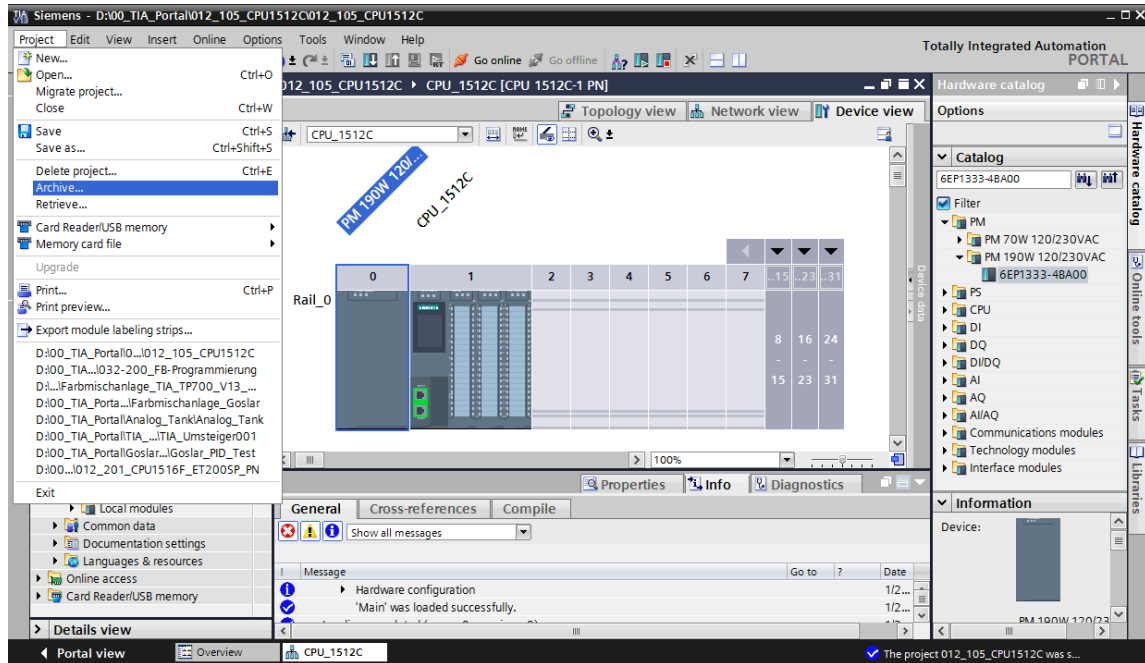
- You can see the downloaded configuration in the project view by double-clicking → "Device configuration". You can switch to the compact view of the simulation by clicking the →  icon in the menu bar.



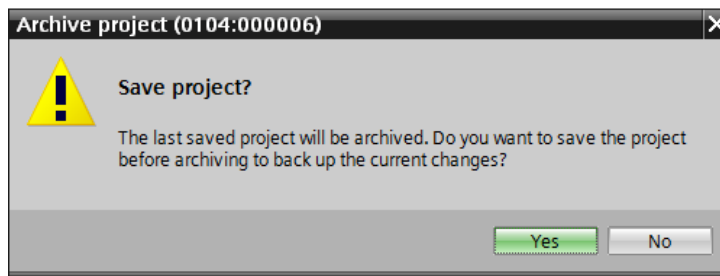
Note: Because this is a simulation, you cannot detect errors in the hardware configuration in this case.

6.9 Archive the project

→ To archive the project, select the → "Archive ..." command in the → "Project" menu.



→ Confirm the prompt to save the project with → "Yes".



→ Select a folder where you want to archive your project and save it as a "TIA Portal project archive" file type. (→ "TIA Portal project archive" → "SCE_EN_012-105_Hardware configuration_S7-1512C..." → "Save")

6.10 Checklist

No.	Description	Completed
1	Project was created	
2	Slot 0: Power module with correct order number	
3	Slot 1: CPU with correct order number	
4	Slot 1: CPU with correct firmware version	
5	Slot 1: Address area of the digital inputs correct	
6	Slot 1: Address area of the digital outputs correct	
7	Slot 1: Address area of the analog inputs correct	
8	Slot 1: Address area of the analog outputs correct	
9	Hardware configuration was compiled without error message	
10	Hardware configuration was downloaded without error message	
11	Project was successfully archived	

7 Exercise

7.1 Task – Exercise

The hardware configuration of the trainer package SIMATIC CPU 1512C PN with Software, PM 1507 and CP 1542-5 (PROFIBUS) is not yet fully complete. Insert the following missing module. Select slot 2 for the communications processor. Set the PROFIBUS address 2 in the properties of the PROFIBUS interface and network this interface with a "PROFIBUS_1" subnet.

- 1X COMMUNICATION PROCESSOR CP 1542-5 FOR CONNECTING SIMATIC S7-1500 TO PROFIBUS DP, DPV1-MASTER OR DP-SLAVE, S7 AND PG/OP COMMUNICATION, TIME SYNCHRONIZATION, DIAGNOSTICS, LESS DATA (ORDER NUMBER: 6GK7542-5FX00-0XE0)

7.2 Planning

Plan the implementation of the task on your own.

7.3 Checklist – Exercise

No.	Description	Completed
1	Slot 2: Communications processor with correct order number	
2	Slot 2: Communications processor with correct firmware version	
3	Slot 2: PROFIBUS interface networked with subnet	
4	Slot 2: PROFIBUS address correct	
5	Hardware configuration was compiled without error message	
6	Hardware configuration was downloaded without error message	
7	Project was successfully archived	

8 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

4

[illegible]

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Continued training

For regional Siemens SCE continued training, get in touch with your regional SCE contact

siemens.com/sce/contact

Additional information regarding SCE

siemens.com/sce

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Description of functions	4
1.1	Brief description	4
1.2	Technology diagram	4
1.3	Switching on	5
1.4	Operating mode selection.....	5
1.5	EMERGENCY STOP.....	5
1.6	Manual mode	5
1.6.1	Retracting and extending the cylinder	5
1.6.2	Conveyor motor in manual mode	5
1.6.3	Initial state.....	5
1.7	Automatic mode.....	6
1.7.1	Starting and stopping.....	6
1.7.2	Conveyor control.....	6
1.7.3	Cylinder control.....	6
1.7.4	Speed control (conveyor speed).....	7
1.7.5	Speed control.....	7
1.8	Indicator lights.....	7
2	Reference list.....	8
3	Description of components of the station	10
3.1	Manual operation	10
3.1.1	Pushbuttons	10
3.1.2	Switches.....	10
3.1.3	Feedback from EMERGENCY STOP pushbutton.....	10
3.2	Sensors.....	10
3.2.1	Position switches	10
3.2.2	Limit switches.....	10
3.2.3	Light barriers / optical sensors.....	10
3.2.4	Metal detection / Inductive sensor	10
3.2.5	Motor speed.....	11
3.3	Actuators.....	11
3.3.1	Conveyor motor	11
3.3.2	Cylinders	11
3.3.3	Displays	11
4	Brief description of the simulation.....	12

PROCESS DESCRIPTION - SORTING STATION

The "Sorting station" example process is described in the following.

1 Description of functions

1.1 Brief description

The automated sorting station (see Figure 1) is used to separate plastic and metal components. A component is fed to the conveyor via a chute. The conveyor starts as soon as the component has been detected. If a component made of metal is on the conveyor, it is detected, transported up to the height of the metal magazine and pushed by a cylinder into the metal magazine. If no metal is detected, the component is made of plastic. The plastic component is transported to the end of the belt, where it falls into the plastic magazine. As soon as a component is sorted, the next component can be fed.

1.2 Technology diagram

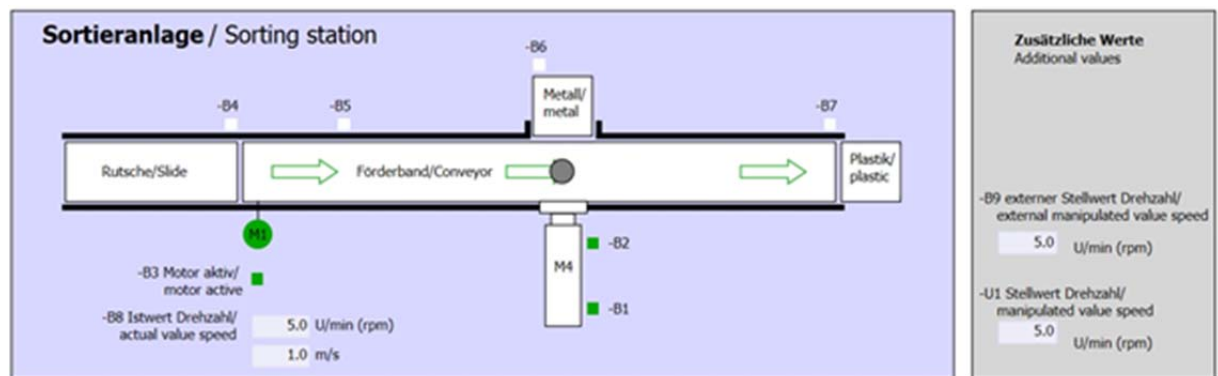


Figure 1: Technology diagram

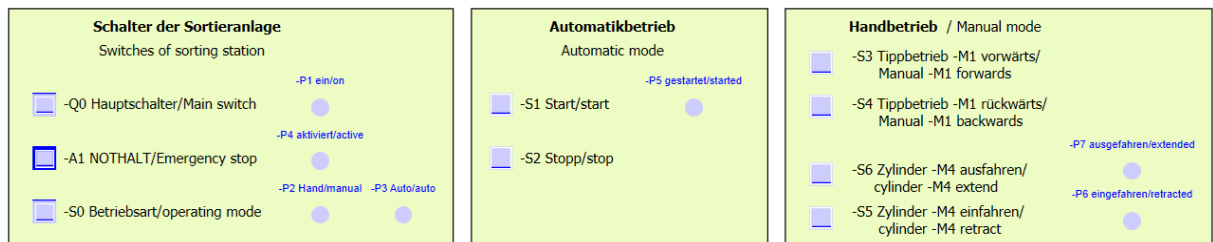


Figure 2: Control panel

1.3 Switching on

The station is switched on with the main switch Q0. Relay K0 (main switch "ON") is energized and provides the supply voltage for the sensors and actuators.

This operating state is indicated by indicator light P1 (main switch on)

1.4 Operating mode selection

Once the station has been switched on, two operating modes are possible: manual mode or automatic mode. The operating mode is selected using switch S0.

The selected operating mode is indicated by indicator lights P2 (manual mode) and P3 (automatic mode).

1.5 EMERGENCY STOP

In the absence of feedback from the EMERGENCY STOP (A1), all drives must be stopped immediately.

When feedback from the EMERGENCY STOP function is present again, the station may only start up again after another start signal.

Activation of the EMERGENCY STOP is indicated by indicator lights P4 (EMERGENCY STOP activated).

1.6 Manual mode

The station is set up in manual mode.

1.6.1 Retracting and extending the cylinder

After pushbutton S5 (cylinder M4 extend) is pressed, cylinder M4 is extended. When the front end position is reached (extended position), the cylinder pauses in this position. After pushbutton S4 is pressed, the cylinder retracts. A change of direction is to be possible at any time. When the two pushbuttons are pressed simultaneously, no motion should take place.

1.6.2 Conveyor motor in manual mode

With pushbutton S3 (pushbutton manual mode conveyor M1 forwards), motor Q1 (conveyor motor M1 forwards fixed speed) is moved forward in manual mode. With pushbutton S4 (pushbutton manual mode conveyor M1 backwards), motor Q2 (conveyor motor M1 backwards fixed speed) is moved backward in manual mode. When the two pushbuttons are pressed simultaneously, no motion should take place.

For safety reasons, only the preset speed may be used here. Output Q3 (conveyor motor M1 variable speed) must therefore be deactivated.

1.6.3 Initial state

At station start or after release of EMERGENCY STOP, the station must be moved in manual mode to a defined operating state (initial state). In the initial state, the conveyor is empty and stopped and the cylinder is retracted.

1.7 Automatic mode

In automatic mode, the station executes the process (see also Brief description).

1.7.1 Starting and stopping

If the station is in the initial state, automatic mode starts when pushbutton S1 (automatic start) is pressed. When pushbutton S2 (automatic stop) is pressed, automatic mode is ended again as soon as the initial state has been reached.

If EMERGENCY STOP has been tripped or the operating mode changed, automatic mode is ended immediately (without return to the initial state).

The current state is indicated by indicator light P6 (automatic mode started).

1.7.2 Conveyor control

If light sensor B4 (chute occupied) detects a component, the conveyor motor starts. The component slides onto the transport conveyor and is further conveyed.

If inductive sensor B5 detected a metal component, this is transported up to light sensor B6 (part in front of cylinder M4). The conveyor is then switched off. As soon as B3 (sensor motor M1 active) no longer supplies a signal, the Cylinder control (see unterhalb) is activated and moves the component into the metal magazine. As soon as the cylinder is retracted again, the sorting station is back in the initial state.

If a metal component was not detected by sensor B5, this is recognized when light sensor B6 (part in front of cylinder M4) is reached. The plastic component is then transported to the end of the conveyor. It is detected there by light sensor B7 and conveyed after a delay time into the plastic magazine at the end of the conveyor.

1.7.3 Cylinder control

If a metal component reaches light sensor B6 (part in front of cylinder M4) and the conveyor has stopped, cylinder M4 moves to the front end position B2 (cylinder M4 extended), thereby pushing the metal component from the conveyor into the metal magazine. Cylinder M4 then moves to the rear end position B1 (cylinder M4 retracted).

1.7.4 Speed control (conveyor speed)

In automatic mode, the motor can be moved at a fixed or variable speed.

Fixed speed requires signal "1" at Q1 "Conveyor motor M1 forwards fixed speed" or Q2 "Conveyor motor M1 backwards fixed speed". For variable speed, Q3 "Conveyor motor M1 variable speed" must be activated and a "manipulated value for motor speed" (analog value +/-10 V corresponds to +/- 50 rpm or 10 m/s) must be specified at U1. Signal "1" must not be present at Q1 "Conveyor motor M1 forwards fixed speed" or Q2 "Conveyor motor M1 backwards fixed speed". Otherwise, U1 has no effect on the speed of the conveyor.

1.7.5 Speed control

A speed control can be integrated for control of the conveyor speed. This uses the speed sensor for evaluating the current speed. A speed of 5 rpm corresponds to a conveyor belt speed of 1 m/s.

1.8 Indicator lights

As soon as relay K0 (main switch "ON") becomes energized, indicator light P1 (main switch on) lights up.

If switch S0 (mode selector manual/automatic) is set to Manual, the indicator light P2 (manual mode) lights up. If switch S0 is set to Automatic, the indicator light P3 (automatic mode) lights up.

If the EMERGENCY STOP function has tripped, P4 (EMERGENCY STOP activated) lights up.

If automatic mode has been selected and the station is in the initial state, P5 (automatic mode started) flashes to signal that automatic mode can be started. As soon as automatic mode has been started, P5 lights up.

Indicator light P6 (cylinder M4 retracted) lights up as soon as end position sensor B1 (sensor cylinder M4 retracted) has been reached. Indicator light P7 (cylinder M4 extended) lights up as soon as cylinder M4 has reached the front end position sensor B2 (sensor cylinder M4 extended). Indicator lights P6 and P7 are not lit if the cylinder is located in neither of the two end positions.

2 Reference list

By default, the S7-1200 has only 14 digital Inputs, 10 digital outputs, 2 analog inputs und 1 analog output. Therefore, the signals shown in the list with blue text are not available for it.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	A1	Return signal emergency stop OK	NC
I 0.1	BOOL	K0	Main switch "ON"	NO
I 0.2	BOOL	S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	S1	Pushbutton automatic start	NO
I 0.4	BOOL	S2	Pushbutton automatic stop	NC
I 0.5	BOOL	B1	Sensor cylinder M4 retracted	NO
I 0.6	BOOL	B2	Sensor cylinder M4 extended	NC
I 0.7	BOOL	B3	Sensor motor M1 active (pulse signal also suitable for positioning)	NO
I 1.0	BOOL	B4	Sensor at chute occupied	NO
I 1.1	BOOL	B5	Sensor metal part	NO
I 1.2	BOOL	B6	Sensor part in front of cylinder M4	NO
I 1.3	BOOL	B7	Sensor part at end of conveyor	NO
I 1.4	BOOL	S3	Pushbutton manual mode conveyor M1 forwards	NO
I 1.5	BOOL	S4	Pushbutton manual mode conveyor M1 backwards	NO
I 1.6	BOOL	S5	Pushbutton manual mode cylinder M4 retract	NO
I 1.7	BOOL	S6	Pushbutton manual mode cylinder M4 extend	NO

DO	Type	Identifier	Function	
Q 0.0	BOOL	Q1	Conveyor motor M1 forwards fixed speed	
Q 0.1	BOOL	Q2	Conveyor motor M1 backwards fixed speed	
Q 0.2	BOOL	Q3	Conveyor motor M1 variable speed	
Q 0.3	BOOL	M2	Cylinder M4 retract	
Q 0.4	BOOL	M3	Cylinder M4 extend	
Q 0.5	BOOL	P1	Display "main switch on"	
Q 0.6	BOOL	P2	Display "MANUAL" mode	
Q 0.7	BOOL	P3	Display "AUTOMATIC" mode	
Q 1.0	BOOL	P4	Display "emergency stop activated"	
Q 1.1	BOOL	P5	Display "automatic mode started"	
Q 1.2	BOOL	P6	Display "cylinder M4 retracted"	
Q 1.3	BOOL	P7	Display "cylinder M4 extended"	
AI	Type	Identifier	Function	
IW 64	INT	B8	Sensor actual value speed of motor +/- 10V	
IW 66	INT	B9	Setpoint specification via potentiometer +/- 10V	
AO	Type	Identifier	Function	
QW 64	INT	U1	Manipulated value speed of motor in 2 directions +/- 10V	

Legend for reference list

DI Digital Input

DO Digital Output

AI Analog Input

AO Analog Output

I Input

Q Output

NC Normally Closed

NO Normally Open

3 Description of components of the station

3.1 Manual operation

3.1.1 Pushbuttons

The utilized pushbuttons can supply either a "0" or "1" signal. Depending on whether you have planned them as normally-closed or normally-open contacts (see Reference list), they supply a "1" or "0" signal when not actuated. The signal changes to "0" or "1" only while the pushbutton is being pressed.

3.1.2 Switches

The utilized switches can also supply either a "0" or "1" signal. Depending on whether you have planned them as normally-closed or normally-open contacts (see Reference list), they supply a "1" or "0" signal when not actuated. The signal changes to "0" or "1" when the switch is actuated. This signal is present as long as the switch is not actuated again.

3.1.3 Feedback from EMERGENCY STOP pushbutton

EMERGENCY STOP pushbuttons are pushbuttons with an additional mechanical lock and are connected to a safety relay. They thus behave like a switch. The EMERGENCY STOP feedback from the safety relay is planned as a normally closed contact for safety reasons. If a wire break occurs, therefore, this feedback is no longer present and the station responds as if an EMERGENCY STOP has tripped.

3.2 Sensors

3.2.1 Position switches

A main switch is actuated to switch on the station. This energizes a relay and supplies the power to the station. A position switch provides feedback on the operation of the relay.

3.2.2 Limit switches

The limit switches supply a signal when the cylinder is either fully retracted or extended. The limit switches are implemented as normally-closed or normally-open contacts.

3.2.3 Light barriers / optical sensors

The light barriers supply a "1" signal as soon as an object is in the sensing range.

3.2.4 Metal detection / Inductive sensor

The inductive sensor supplies a "1" signal as soon as a metallic object enters its sensing range. In the case of non-metallic objects, the signal remains at "0".

3.2.5 Motor speed

The motor speed is recorded by an incremental encoder at the conveyor motor and provided as an analog value via a transducer. The speed falls within the range from -50 rpm to 50 rpm. That corresponds to a conveyor belt speed of -10 m/s to +10 m/s.

In addition, pulses are received at "Sensor conveyor motor M1 active" that can also be used for positioning. The resolution is 20 pulses per total conveyor belt length (10 m).

3.3 Actuators

3.3.1 Conveyor motor

The conveyor motor drives the conveyor belt. It has multiple signal combinations so that the conveyor belt can be moved at fixed or variable speed in both directions.

Fixed speed requires signal "1" at Q1 "Conveyor motor M1 forwards fixed speed" or Q2 "Conveyor motor M1 backwards fixed speed". For variable speed, Q3 "Conveyor motor M1 variable speed" must be activated and a "manipulated value for motor speed" (analog value +/-10 V corresponds to +/- 50 rpm or 10 m/s) must be specified at U1. Signal "1" must not be present at Q1 "Conveyor motor M1 forwards fixed speed" or Q2 "Conveyor motor M1 backwards fixed speed". Otherwise, U1 has no effect. Simultaneous activation of signals Q1 and Q2 causes the conveyor to stop and must be prevented by the control program.

3.3.2 Cylinders

Cylinder M4 is controlled using two separate signals. Activation of one signal (M3) causes the cylinder to extend and activation of the other signal (M2) causes the cylinder to retract. The signals must not be activated simultaneously, otherwise an undefined state occurs and the cylinder pauses at its position. This must be prevented by the control program.

3.3.3 Displays

All indicator lights are located on the control panel. If signal "1" is present, these indicator lights illuminate.

4 Brief description of the simulation

The simulation of the sorting station consists of 9 diagrams. The 01_Operating screen diagram is important for operation (see Figure 3), which contains the control panel and a representation of the station.

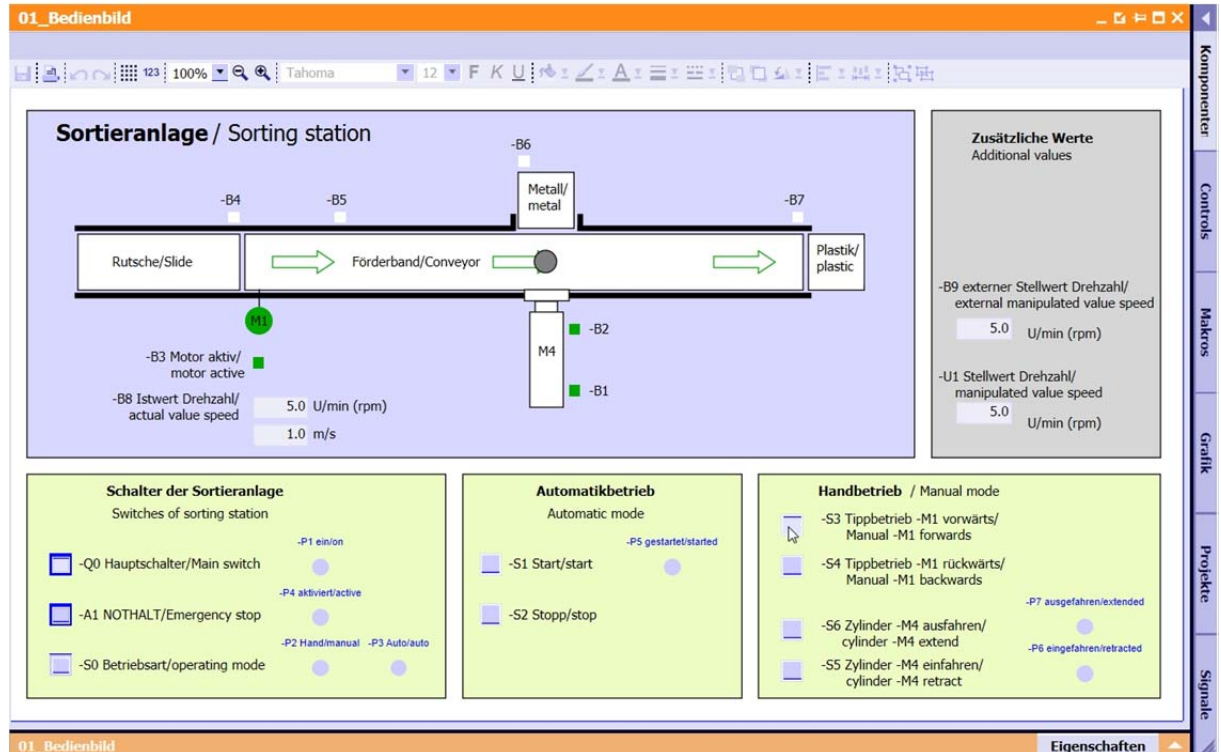


Figure 3: The operating screen

Figure 4 shows the 02_SimControl diagram. It allows important simulation settings to be made. The first settings affect the creation of the components. Here, you can select between automatic and manual creation of components. With automatic creation of components, a new component is always created and sent to the station when the previous component has been sorted. A single component is created with manual creation of components. The next settings allow you to specify whether a metal component or plastic component will be created. The following selections are available: Produce only metal components, Produce only plastic components and Randomly produce metal or plastic components. Only one of the three options should be selected.

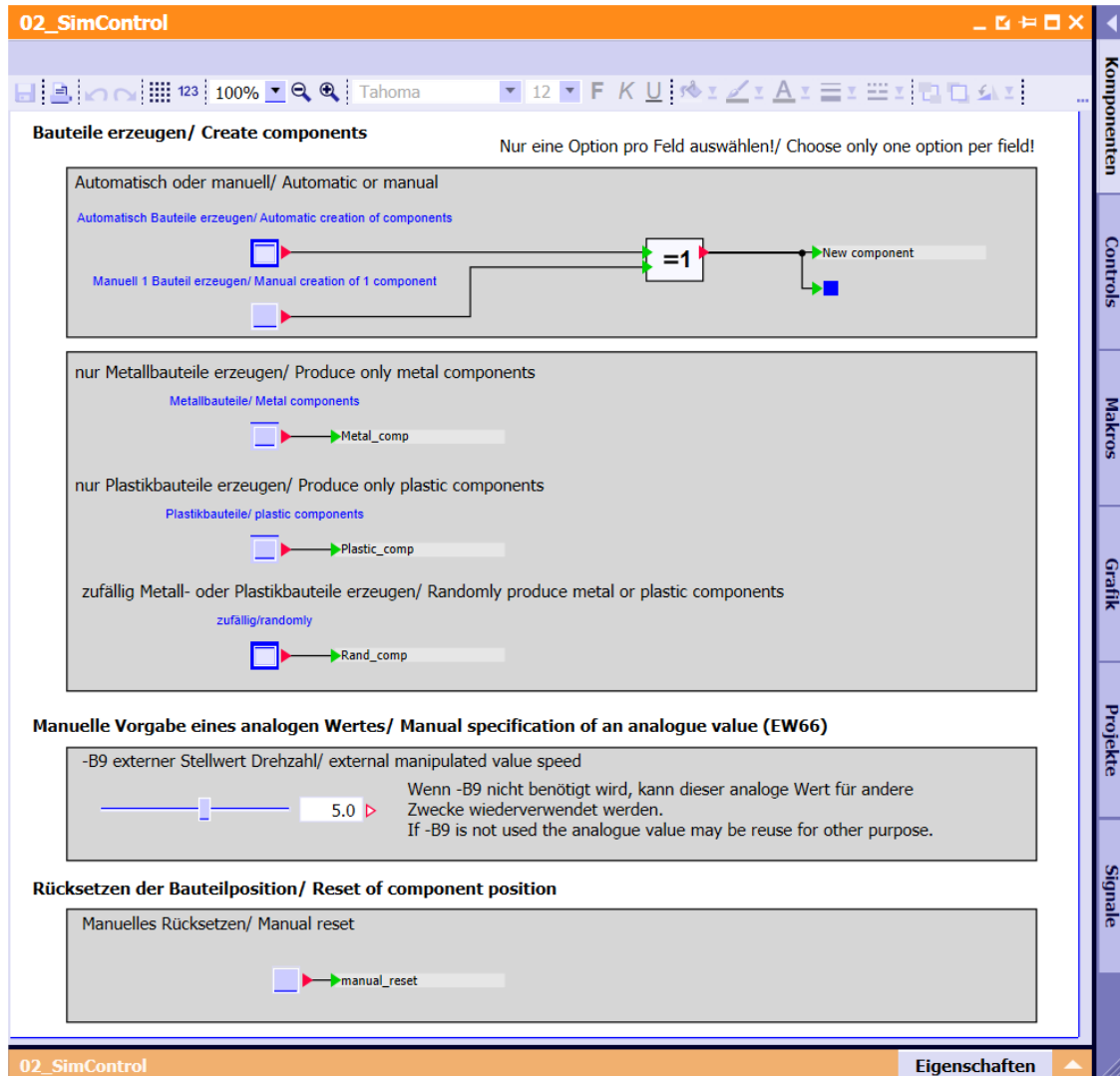


Figure 4: Simulation control

In the "Manual specification of an analog value" area, a value between -50 and +50 can be set for input word IW 66 (see Reference list). This corresponds to an input voltage of ± 10 V. This value is then converted to a digital value between -27648 and +27648 and is thus available as an analog input value.

The last setting concerns the manual resetting of the current component. This resets the position of the component and a new component can be created.

5

[illegible]

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

6

Continued training

For regional Siemens SCE continued training, get in touch with your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	4
2	Prerequisite.....	4
3	Theory.....	4
3.1	Operating system and application program	4
3.2	Organization blocks.....	5
3.3	Process image and cyclic program processing.....	6
3.4	Functions.....	8
3.5	Function blocks and instance data blocks.....	9
3.6	Global data blocks.....	10
3.7	Library-compatible code blocks.....	11
3.8	Programming languages	12
4	Task	13
5	Planning.....	13
5.1	EMERGENCY STOP	13
5.2	Manual mode – Conveyor motor in manual mode	13
6	Structured step-by-step instructions	14
6.1	Retrieve an existing project.....	14
6.2	Create a new tag table	15
6.3	Create new tags within a tag table	17
6.4	Import "Tag_table_sorting_station"	18
6.5	Create function FC1 "MOTOR_MANUAL" for the conveyor motor in manual mode	22
6.6	Define the interface of function FC1 "MOTOR_MANUAL"	24
6.7	Program FC1: MOTOR_MANUAL	27
6.8	Program organization block OB1 – Control of the forward belt tracking in manual mode	34
6.9	Program organization block OB1 – Control of the backward belt tracking in manual mode.....	39
6.10	Save and compile the program	41
6.11	Download the program	42
6.12	Monitor program blocks.....	43
6.13	Archive the project.....	45
7	Checklist	46
8	Exercise	47
8.1	Task – Exercise.....	47
8.2	Planning	47
8.3	Checklist – Exercise	48
9	Additional information	49

BASICS OF FC PROGRAMMING

1 Goal

In this chapter, you will get to know the basic elements of a control program – the **organization blocks (OBs)**, **functions (FCs)**, **function blocks (FBs)** and **data blocks (DBs)**. In addition, you will be introduced to **library-compatible** function and function block programming. You will get to know the **Function Block Diagram (FBD)** programming language and use it to program a function (FC1) and an organization block (OB1).

2 Prerequisite

This chapter builds on the hardware configuration of SIMATIC S7 CPU1516F-3 PN/DP. However, other hardware configurations that have digital input and output cards can be used. You can use the following project for this chapter, for example:

SCE_EN_012_101__Hardware_Configuration_CPU1516F.zap13

3 Theory

3.1 Operating system and application program

Every controller (CPU) contains an **operating system**, which organizes all functions and processes of the CPU that are not associated with a specific control task. The tasks of the operating system include the following:

- Performing a warm restart
- Updating the process image of the inputs and outputs
- Cyclically calling the user program
- Detecting interrupts and calling interrupt OBs
- Detecting and handling errors
- Managing memory areas

The operating system is an integral component of the CPU and comes pre-installed.

The **user program** contains all functions that are necessary for executing your specific automation task. The tasks of the user program include the following:

- Checking the basic requirements for a warm restart using startup OBs
- Processing of process data, i.e. activation of output signals as a function of the input signal states
- Reaction to interrupts and interrupt inputs
- Error handling during normal program execution

3.2 Organization blocks

Organization blocks (OBs) form the interface between the operating system of the controller (CPU) and the application program. They are called from the operating system and control the following operations:

- Cyclic program processing (e.g. OB1)
- Startup characteristics of the controller
- Interrupt-driven program processing
- Error handling

A project must have **an organization block for cyclic program processing** at a minimum. An OB is called by a **start event** as shown in Figure 1. In addition, the individual OBs have defined priorities so that, for example, an OB82 for error handling can interrupt the cyclic OB1.

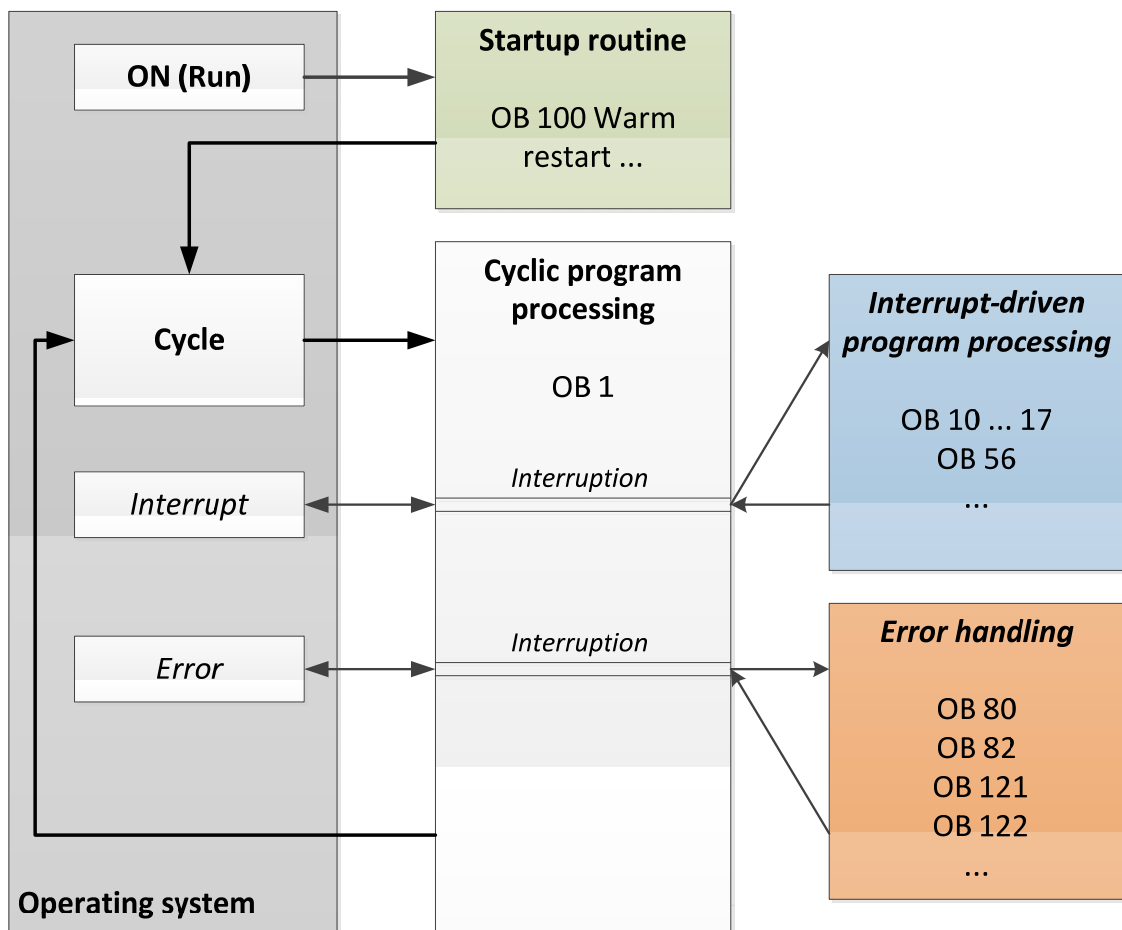


Figure 1: Start events in the operating system and OB call

When a start event occurs, the following reactions are possible:

- If an OB has been assigned to the event, this event triggers the execution of the assigned OB. If the priority of the assigned OB is greater than the priority of the OB that is currently being executed, it is executed immediately (interrupt). If not, the assigned OB waits until the higher-priority OB has been completely executed.
- If an OB is not assigned to the event, the default system reaction is performed.

Table 1 gives a couple of examples of start events for a SIMATIC S7-1500, their possible OB number(s) and the default system reaction in the event the organization block is not present in the controller.

Start event	Possible OB numbers	Default system reaction
Startup	100, ≥ 123	Ignore
Cyclic program	1, ≥ 123	Ignore
Time-of-day interrupt	10 to 17, ≥ 123	-
Update interrupt	56	Ignore
Scan cycle monitoring time exceeded once	80	STOP
Diagnostic interrupt	82	Ignore
Programming error	121	STOP
I/O access error	122	Ignore

Table 1: OB numbers for various start events

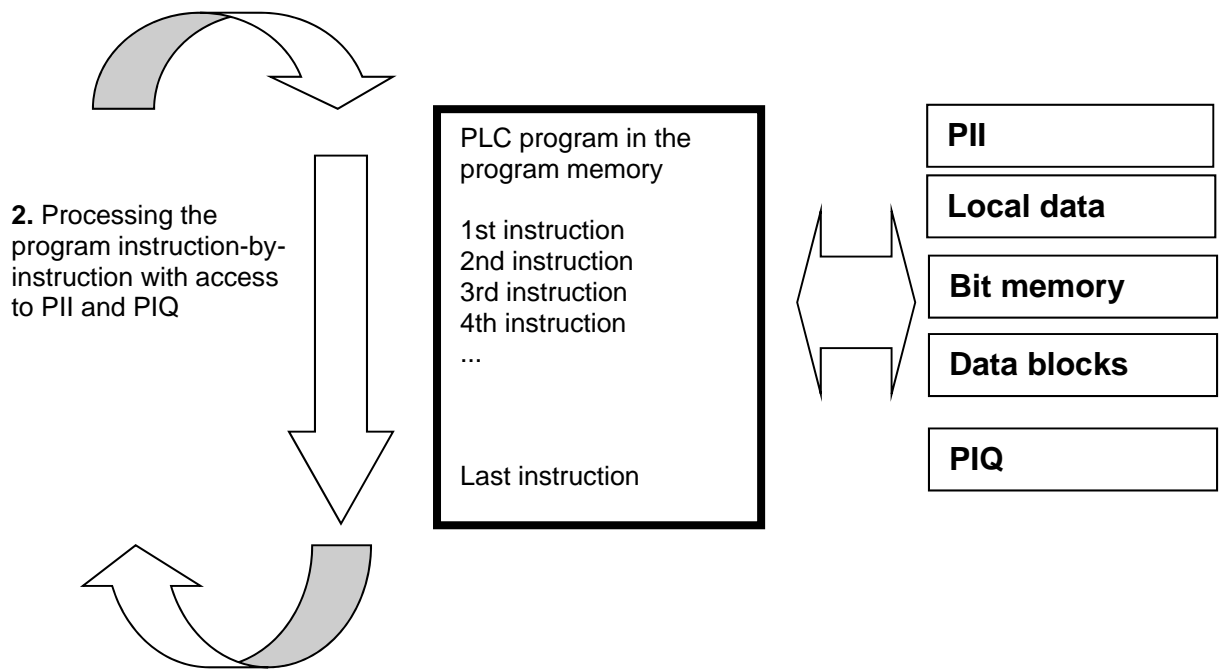
3.3 Process image and cyclic program processing

When the cyclic user program addresses the inputs (I) and outputs (O), it does not query the signal states directly from the input/output modules. Instead, it accesses a memory area of the CPU. This memory area contains an image of the signal states and is called the **process image**.

The cyclic program processing sequence is as follows:

1. At the start of the cyclic program, a query is sent to determine whether or not the individual inputs are energized. This status of the inputs is stored in the **process image of the inputs (PII)**. In doing so, the information 1 or "High" is stored for energized inputs and the information 0 or "Low" for de-energized inputs.
2. The CPU now executes the program stored in the cyclic organization block. For the required input information, the CPU accesses the previously read **process image of the inputs (PII)** and the results of logic operation (RLOs) are written to a so-called **process image of the outputs (PIQ)**.
3. At the end of the cycle, the **process image of the outputs (PIQ)** is transferred as the signal state to the output modules and these are energized or de-energized. The sequence then continues again with Item 1.

1. Save status of inputs in PII.



3. Transfer status from the PIQ to the outputs.

Figure 2: Cyclic program processing

Note: The time the CPU needs for this sequence is called cycle time. This depends, in turn, on the number and type of instructions and the processor performance of the controller.

3.4 Functions

Functions (FCs) are logic blocks without memory. They **have no data memory** in which values of block parameters can be stored. Therefore, all interface parameters must be connected when a function is called. To store data permanently, global data blocks must be created beforehand.

A function contains a program that is executed whenever the function is called from another code block.

Functions can be used, for example, for the following purposes:

- Math functions – that return a result dependent on input values.
- Technological functions – such as individual controls with binary logic operations.

A function can also be called several times at different points within a program.

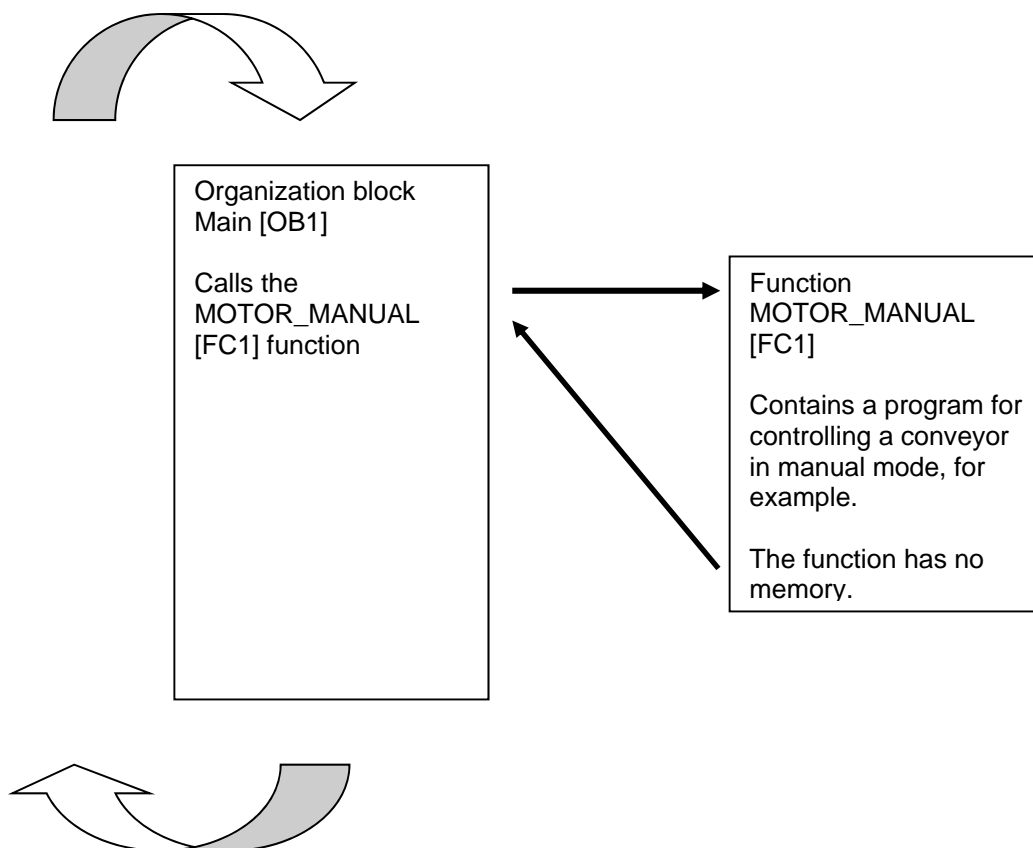


Figure 3: Function with call from organization block Main [OB1]

3.5 Function blocks and instance data blocks

Function blocks are code blocks that store their input, output and in-out tags as well as static tags permanently in instance data blocks, so that they **are available after the block has been executed**. For this reason, they are also referred to as blocks with "memory".

Function blocks can also operate with temporary tags. Temporary tags are not stored in the instance DB, however. Instead, they are only available for one cycle.

Function blocks are used for tasks that cannot be implemented with functions:

- Whenever timers and counters are required in the blocks.
- Whenever information must be saved in the program, such as pre-selection of the operating mode with a button.

Function blocks are always executed when called from another code block. A function block can also be called several times at different points within a program. This facilitates the programming of frequently recurring complex functions.

A call of a function block is referred to as an instance. Each instance of a function block is assigned a memory area that contains the data that the function block uses. This memory is made available by data blocks created automatically by the software.

It is also possible to provide memory for multiple instances in one data block in the form of a **multi-instance**. The maximum size of instance data blocks varies depending on the CPU. The tags declared in the function block determine the structure of the instance data block.

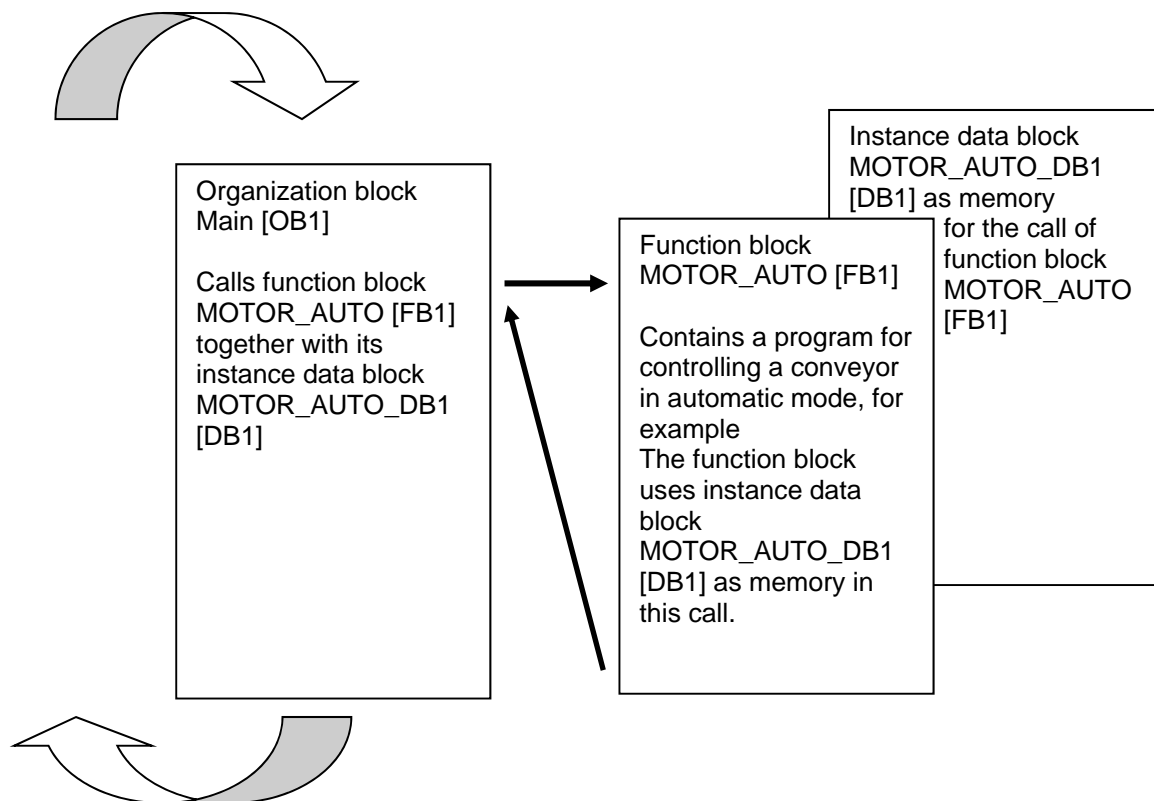


Figure 4: Function block and instance with call from organization block Main [OB1]

3.6 Global data blocks

In contrast to logic blocks, data blocks contain no instructions. Rather, they serve as memory for user data.

Data blocks thus contain variable data that is used by the user program. You can define the structure of global data blocks as required.

Global data blocks store data that can be used **by all other blocks** (see Figure 5). Only the associated function block should access instance data blocks. The maximum size of data blocks varies depending on the CPU.

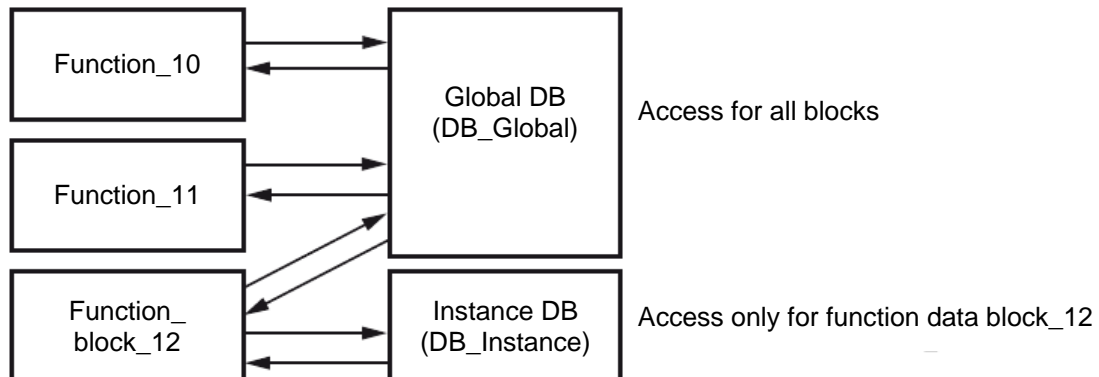


Figure 5: Difference between global DB and instance DB.

Application examples for **global data blocks** are:

- Saving of information about a storage system. "Which product is located where?"
- Saving of recipes for particular products.

3.7 Library-compatible code blocks

A user program can be created with linear or structured programming. **Linear programming** writes the entire user program in the cycle OB, but is only suitable for very simple programs for which other less expensive control systems, such as LOGO!, can now be used.

Structured programming is always recommended for more complex programs. Here, the overall automation task can be broken down into small sub-tasks in order to implement a solution for them in functions and function blocks.

In this case, library-compatible logic blocks should be created preferentially. This means that the input and output parameters of a function or function block are defined generally and only supplied with the current global tags (inputs/outputs) when the block is used.

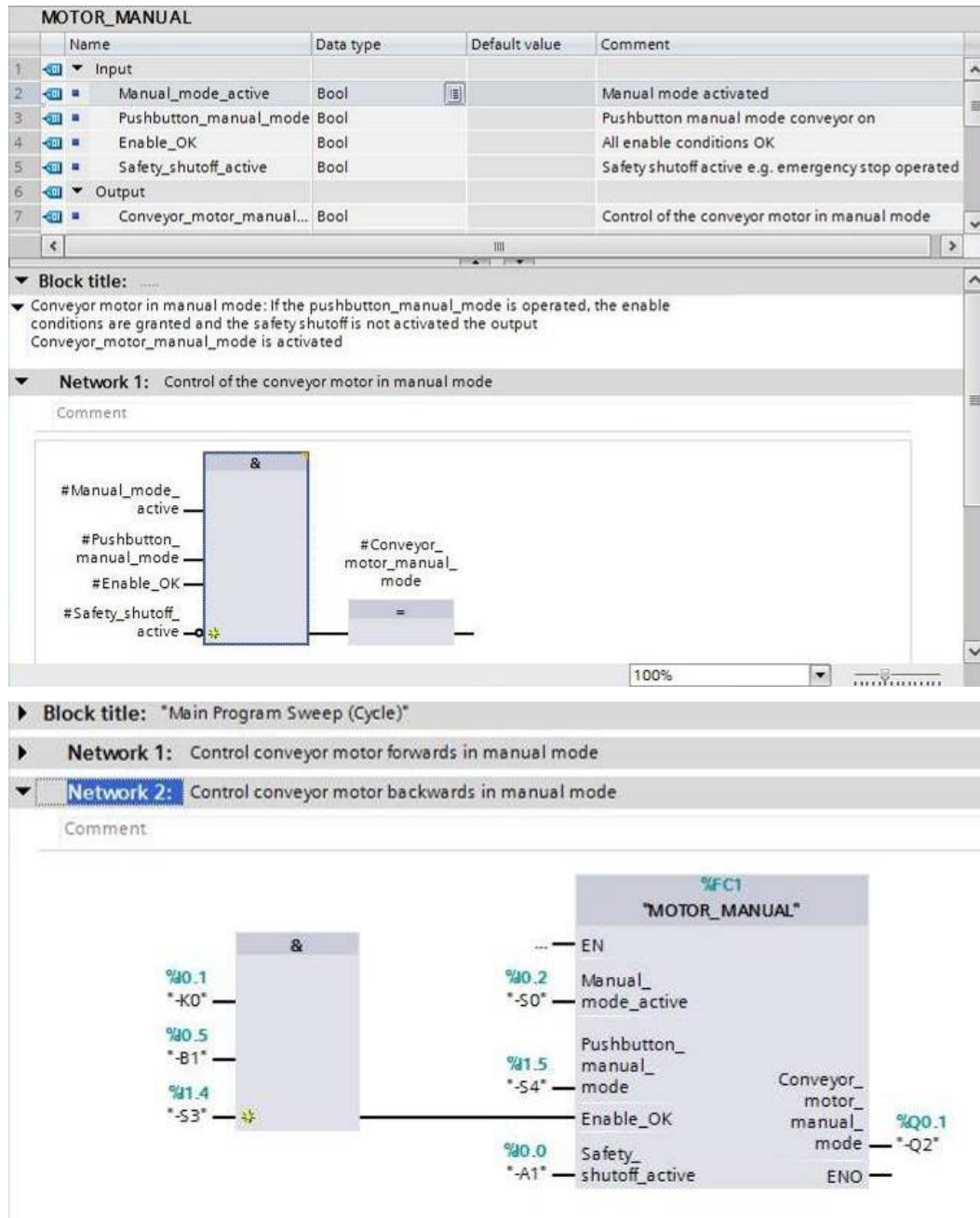


Figure 6: Library-compatible function with call in OB1

3.8 Programming languages

The available programming languages for programming functions are Function Block Diagram (FBD), Ladder Logic (LAD), Statement List (STL) and Structured Control Language (SCL). For function blocks, the GRAPH programming language is additionally available for programming graphical step sequences.

The **Function Block Diagram (FBD)** programming language will be presented in the following.

FBD is a graphical programming language. The representation is based on electronic switching systems. The program is mapped in networks. A network contains one or more logic operation paths. Binary and analog signals are linked by boxes. The graphical logic symbols known from Boolean algebra are used to represent the binary logic.

You can use binary functions to query binary operands and to logically combine their signal states. The following instructions are examples of binary functions: "AND operation", "OR operation" and "EXCLUSIVE OR operation". These are shown in Figure 7.

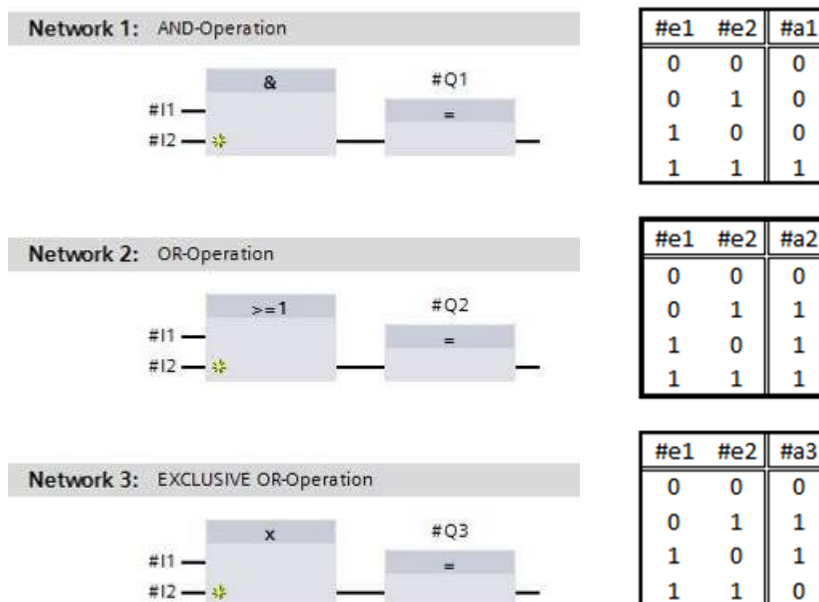


Figure 7: Binary functions in FBD and associated logic table

You can thus use simple instructions, for example, to control binary outputs, evaluate edges and execute jump functions in the program.

Program elements such as IEC timers and IEC counters provide complex instructions.

The empty box serves as a placeholder in which you can select the required instruction.

Enable input EN (enable)/ Enable output ENO (enable output) mechanism:

- An instruction without EN/ENO mechanism is executed independent of the signal state at the box inputs.
- Instructions with EN/ENO mechanism are only executed if enable input "EN input has signal state "1". When the box is processed correctly, enable output "ENO" has signal state "1". As soon as an error occurs during the processing, the "ENO" enable output is reset. If enable input EN is not connected, the box is always executed.

4 Task

The following functions of the sorting station process description will be planned, programmed and tested in this chapter:

- Manual mode – Conveyor motor in manual mode

5 Planning

The programming of all functions in OB1 is not recommended for reasons of clarity and reusability. The majority of the program code will therefore be moved into functions (FCs) and function blocks (FBs). The decision on which functions are to be moved to FCs and which is to run in OB 1 is planned below.

5.1 EMERGENCY STOP

The EMERGENCY STOP does not require a separate function. Just like the operating mode, the current state of the EMERGENCY STOP relay can be used directly at the blocks.

5.2 Manual mode – Conveyor motor in manual mode

Manual mode of the conveyor motor is to be encapsulated in a function (FC) "MOTOR_MANUAL". On the one hand, this preserves the clarity of OB1. On the other hand, it enables reuse if another conveyor belt is added to the station. Table 2 lists the planned parameters.

Input	Data	Comment
Manual_mode_active	BOOL	Manual mode activated
Pushbutton_manual_mode	BOOL	Pushbutton manual mode conveyor on
Enable_OK	BOOL	All enable conditions OK
Safety_shutoff_active	BOOL	Safety shutoff active, e.g. emergency stop pressed
Output		
Conveyor_motor_manual_mode	BOOL	Control of the conveyor motor in manual mode

Table 2: Parameters for FC "MOTOR_MANUAL"

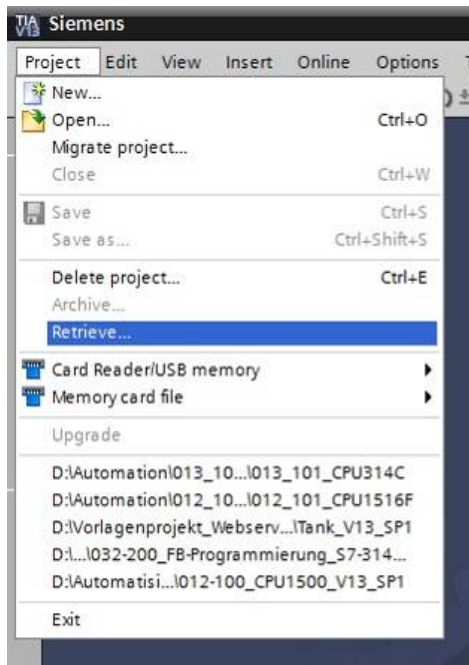
Output Conveyor_motor_manual_mode is ON as long as Pushbutton_manual_mode is pressed, manual mode is activated, the enable conditions are OK and the safety shutoff is not active.

6 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

6.1 Retrieve an existing project

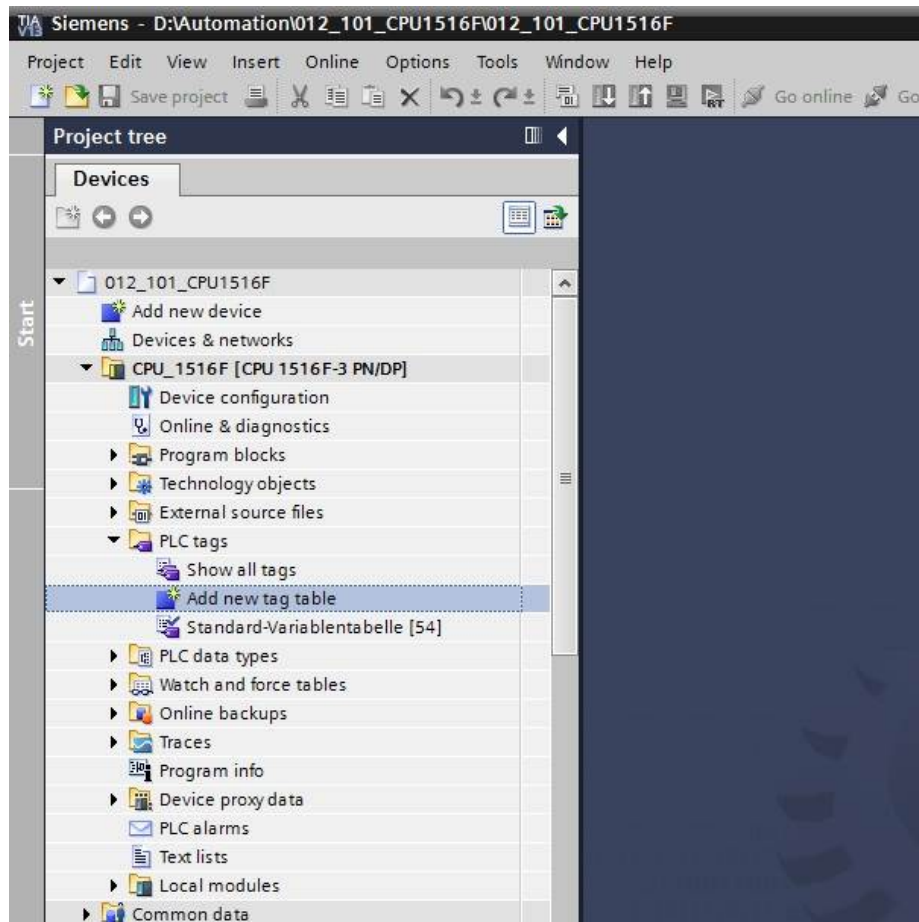
- Before we can start programming the function (FC) "MOTOR_MANUAL", we need a project with a hardware configuration (e.g. SCE_EN_012_101_Hardware_Configuration_S7-1516F_R1502.zap). To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view Confirm your selection with Open. (→ Project → Retrieve → Select a .zap archive → Open)



- The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK". (→ Target directory → OK)

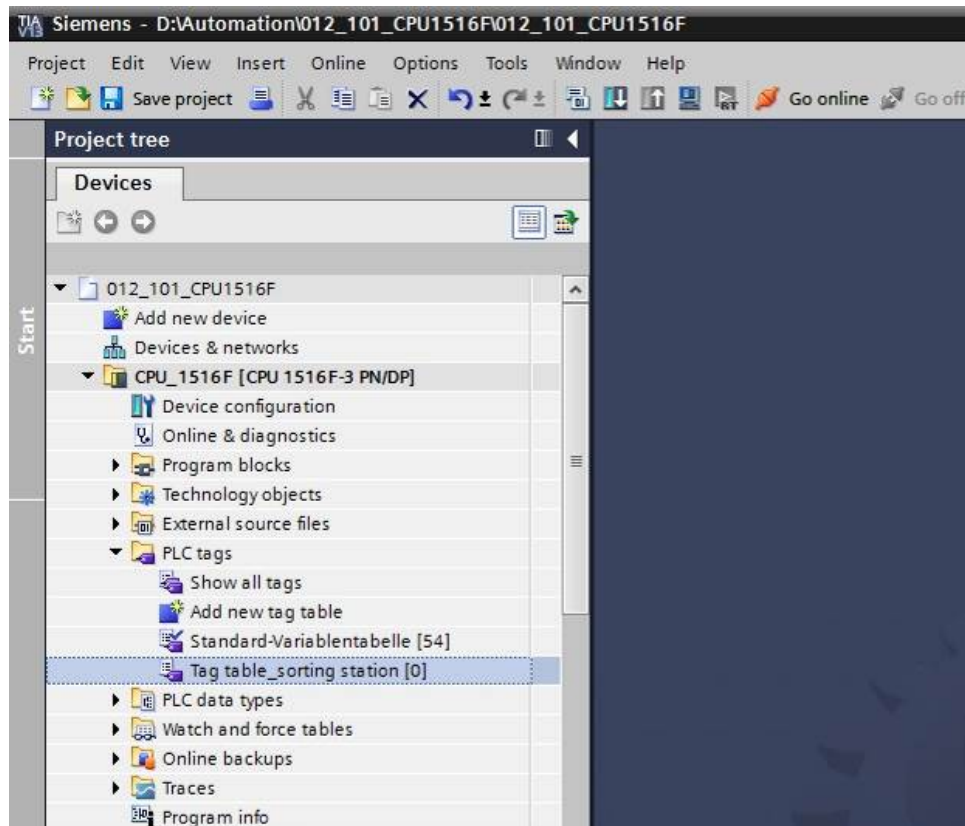
6.2 Create a new tag table

- In the project view, navigate to the → PLC tags of your controller and create a new tag table by double-clicking → Add new tag table.

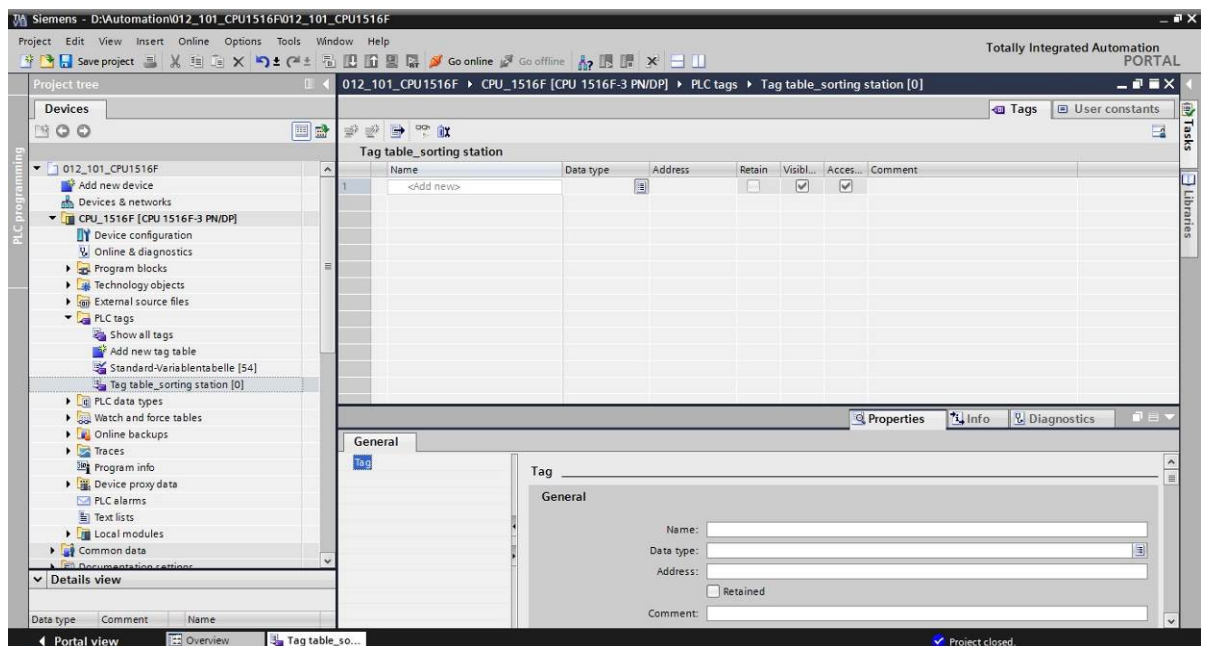


- Rename the tag table you just created as "Tag_table_sorting_station" (→ right-click "Tag_table_1" → "Rename" → Tag_table_sorting_station).

6

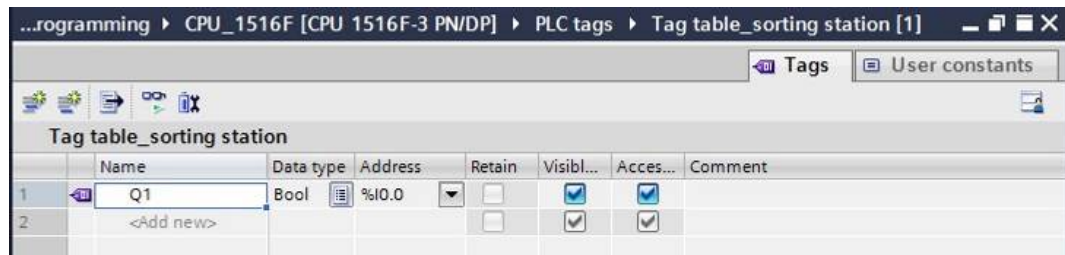


→ Open this tag table with a double-click. (→ Tag_table_sorting_station)

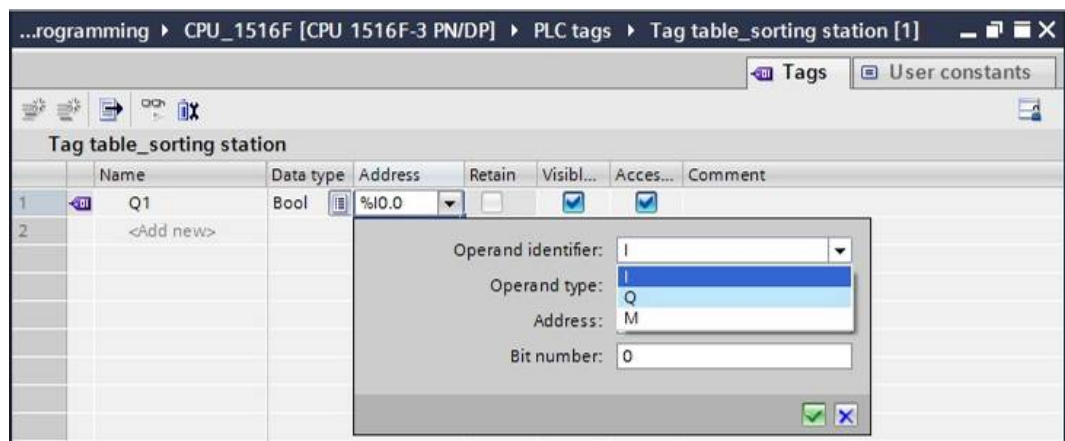


6.3 Create new tags within a tag table

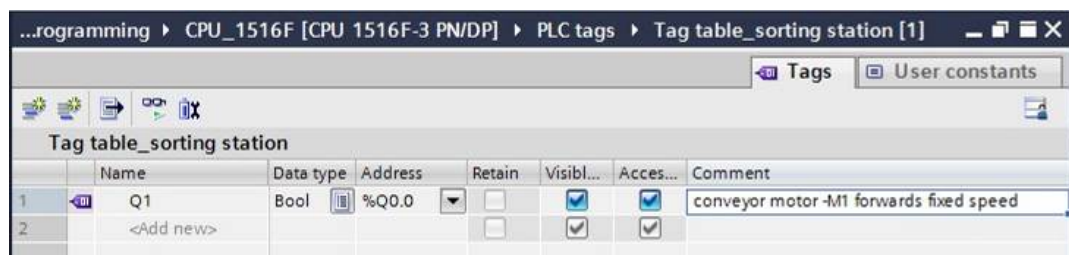
- Add the name Q1 and confirm the entry with the Enter key. If you have not yet created additional tags, TIA Portal now automatically assigns data type "Bool" and address %I0.0 (I 0.0) (→ <Add> → Q1 → Enter).



- Change the address to %Q0.0 (Q 0.0) by entering this directly or by clicking the drop-down arrow to open the Addressing menu, changing the operand identifier to Q and confirming with Enter or by clicking the check mark. (→ %I0.0 → Operand identifier → Q → ☒)



- Enter the "Conveyor motor M1 forwards fixed speed" comment for the tag.



→ Add a new Q2 tag in line 2. TIA Portal has automatically assigned the same data type as in line 1 and has incremented the address by 1 to %Q0.1 (Q0.1). Enter the comment "Conveyor motor M1 backwards fixed speed".

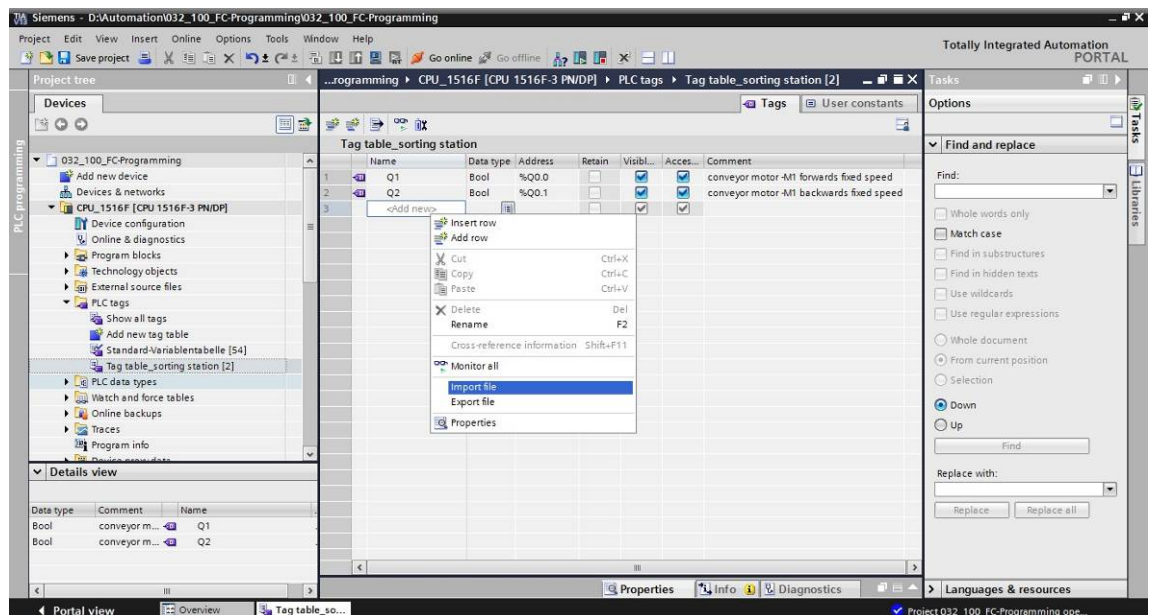
(→ <Add> → Q2 → Enter → Comment → Conveyor motor M1 backwards fixed speed)

	Name	Data type	Address	Retain	Visibl...	Acces...	Comment
1	Q1	Bool	%Q0.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 forwards fixed speed
2	Q2	Bool	%Q0.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 backwards fixed speed
3	<Add new>				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

6.4 Import "Tag_table_sorting_station"

→ To insert an existing symbol table, right-click on an empty field of the created "Tag_table_sorting_station". Select "Import file" in the shortcut menu.

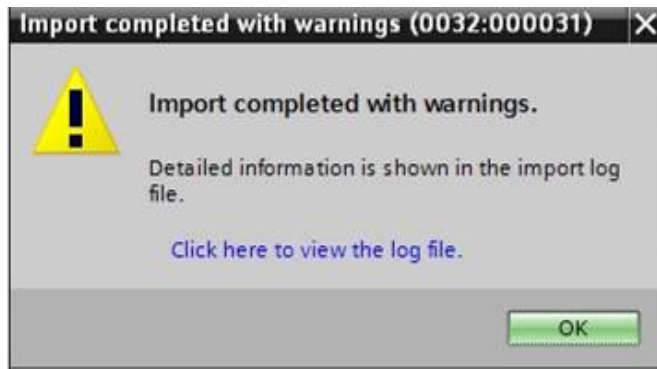
(→ Right-click in an empty field of the tag table → Import file)



→ Select the desired symbol table (e.g. in .xlsx format) and confirm the selection with "Open".

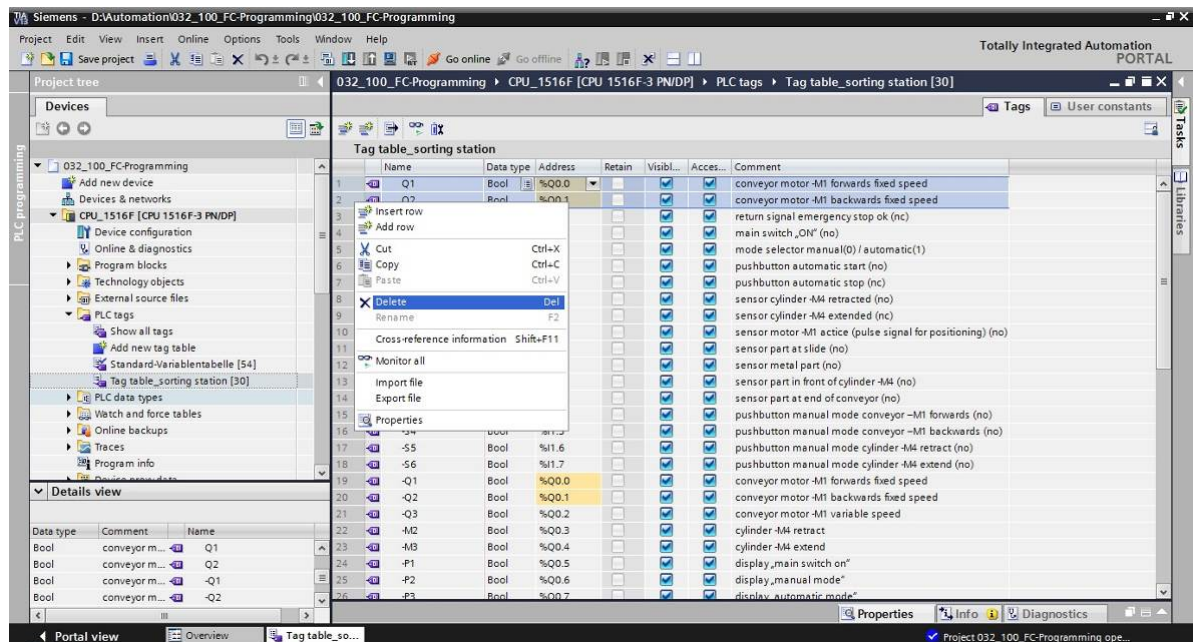
(→ SCE_EN_020-100_Tag_table_sorting_station... → Open)

→ When the import is finished, you will see a confirmation window and have an opportunity to view the log file for the import. Click → OK.



SCE Training Curriculum | TIA Portal Module 032-100, Edition 02/2016 | Digital Factory, DF FA

- You can see that some addresses have been highlighted in orange. These are duplicate addresses and the names of the associated tags have been numbered automatically to avoid confusion.
- Delete the duplicate tags by selecting the lines and pressing the Del key on your keyboard or by selecting "Delete" in the shortcut menu.
- (→ Right-click on selected tags → Delete)



→ You now have a complete symbol table of the digital inputs and outputs in front of you. Save your project under the name 032-100_FCProgramming.

(→ Project → Save as ...→032-100_FCProgramming → Save)


The screenshot shows the Siemens TIA Portal interface. The 'File' menu is open, and the 'Save as...' option is highlighted. The main window displays the 'Tag table_sorting station' table, which lists 26 digital inputs and outputs. The table has columns for Name, Data type, Address, Retain, Visibl., Acces., and Comment. The comments describe various signals such as 'return signal emergency stop ok (nc)', 'main switch „ON“ (no)', 'mode selector manual(0) / automatic(1)', 'pushbutton automatic start (no)', 'pushbutton automatic stop (nc)', 'sensor cylinder -M4 retracted (no)', 'sensor cylinder -M4 extended (nc)', 'sensor motor -M1 actice (pulse signal for ...)', 'sensor part at slide (no)', 'sensor metal part (no)', 'sensor part in front of cylinder -M4 (no)', 'sensor part at end of conveyor (no)', 'pushbutton manual mode conveyor -M1...', 'pushbutton manual mode cylinder -M4 re...', 'pushbutton manual mode cylinder -M4 ex...', 'conveyor motor -M1 forwards fixed speed', 'conveyor motor -M1 backwards fixed speed', 'conveyor motor -M1 variable speed', 'cylinder -M4 retract', 'cylinder -M4 extend', 'display „main switch on“', 'display „manual mode“', 'display „automatic mode“', 'display „emergency stop activated“', and 'display „automatic mode started“'.

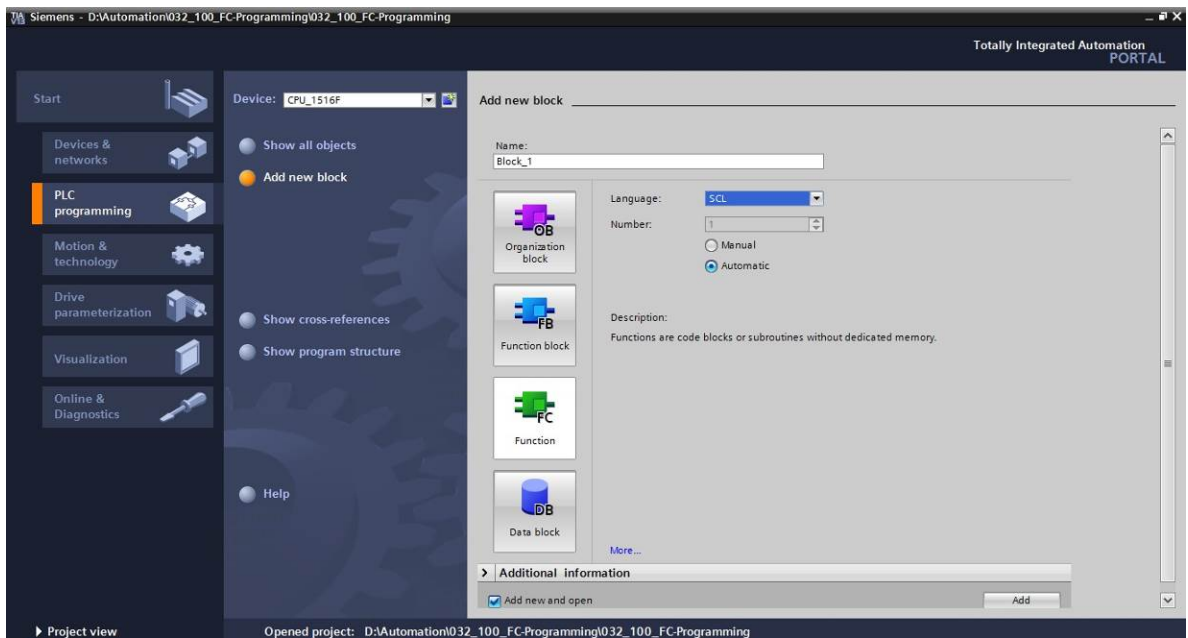
Name	Data type	Address	Retain	Visibl.	Acces.	Comment
1	Bool	%I0.0				return signal emergency stop ok (nc)
2	Bool	%I0.1				main switch „ON“ (no)
3	Bool	%I0.2				mode selector manual(0) / automatic(1)
4	Bool	%I0.3				pushbutton automatic start (no)
5	Bool	%I0.4				pushbutton automatic stop (nc)
6	Bool	%I0.5				sensor cylinder -M4 retracted (no)
7	Bool	%I0.6				sensor cylinder -M4 extended (nc)
8	Bool	%I0.7				sensor motor -M1 actice (pulse signal for ...)
9	Bool	%I1.0				sensor part at slide (no)
10	Bool	%I1.1				sensor metal part (no)
11	Bool	%I1.2				sensor part in front of cylinder -M4 (no)
12	Bool	%I1.3				sensor part at end of conveyor (no)
13	Bool	%I1.4				pushbutton manual mode conveyor -M1...
14	Bool	%I1.5				pushbutton manual mode cylinder -M4 re...
15	Bool	%I1.6				pushbutton manual mode cylinder -M4 ex...
16	Bool	%I1.7				conveyor motor -M1 forwards fixed speed
17	Bool	%Q0.0				conveyor motor -M1 backwards fixed speed
18	Bool	%Q0.1				conveyor motor -M1 variable speed
19	Bool	%Q0.2				cylinder -M4 retract
20	Bool	%Q0.3				cylinder -M4 extend
21	Bool	%Q0.4				display „main switch on“
22	Bool	%Q0.5				display „manual mode“
23	Bool	%Q0.6				display „automatic mode“
24	Bool	%Q0.7				display „emergency stop activated“
25	Bool	%Q1.0				display „automatic mode started“
26	Bool	%Q1.1				

6

6.5 Create function FC1 "MOTOR_MANUAL" for the conveyor motor in manual mode

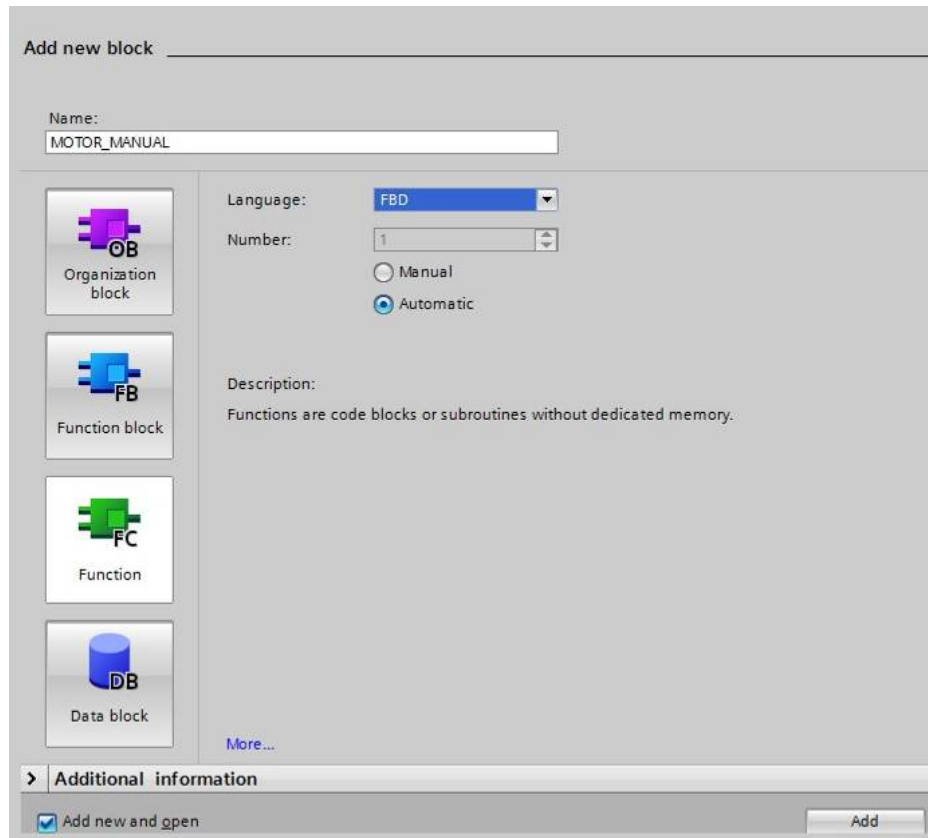
→ In the PLC programming section of the portal view, click "Add new block" to create a new function.

(→ PLC programming → Add new block → )



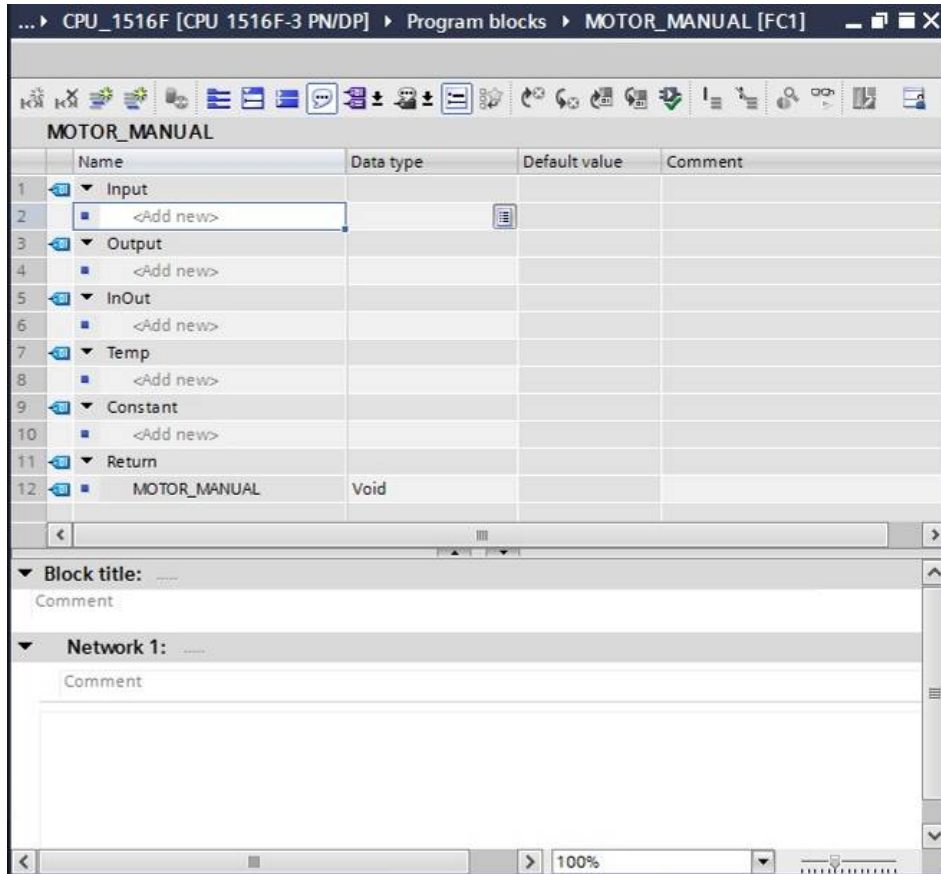
→ Rename your new block to: "MOTOR_MANUAL", set the language to FBD and keep automatic assignment of the number. Select the "Add new and open" check box. You are then taken automatically to your created function block in the project view. Click "Add".

(→ Name: MOTOR_MANUAL → Language: FBD → Number: Automatic → ☒ Add new and open → Add)



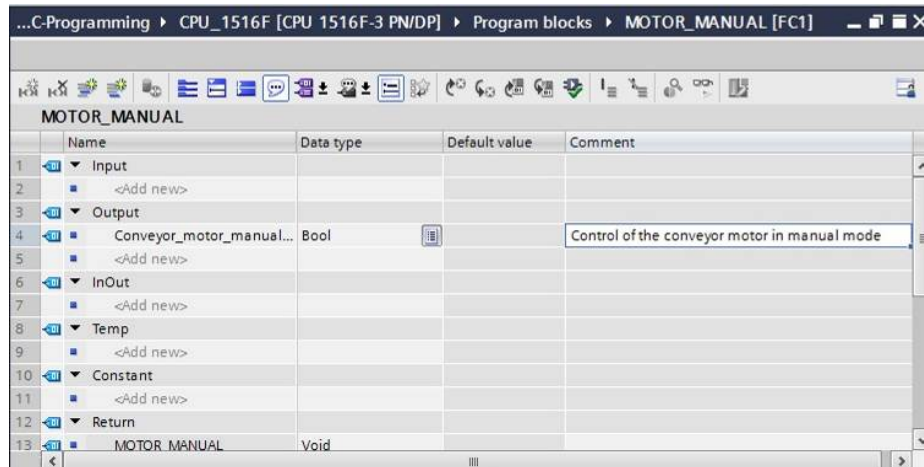
6.6 Define the interface of function FC1 "MOTOR_MANUAL"

- If you selected "Add new and open", the project view opens with a window for creating the block you just added.
- You can find the interface description of your function in the upper section of your programming view.



- A binary output signal is needed for controlling the conveyor motor. For this reason, we first create local output tag #Conveyor_motor_manual_mode of the "Bool" type. Enter comment "Control of the conveyor motor in manual mode" for the parameter.

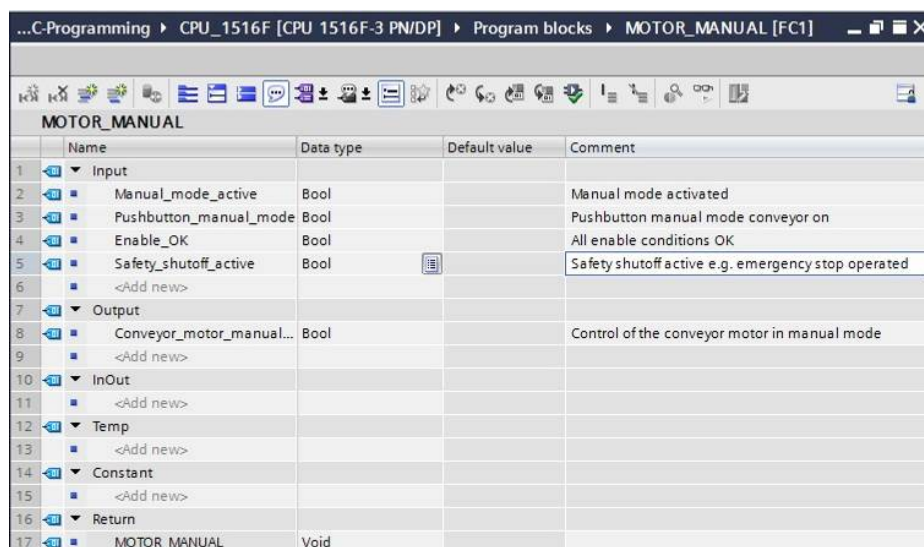
(→ Output: Conveyor_motor_manual_mode → Bool → Control of the conveyor motor in manual mode)



- Add parameter #Manual_mode_active as the input interface under Input and confirm the entry with the Enter key or by exiting the entry field. Data type "Bool" is assigned automatically. This will be retained. Next, enter the associated comment "Manual mode activated".

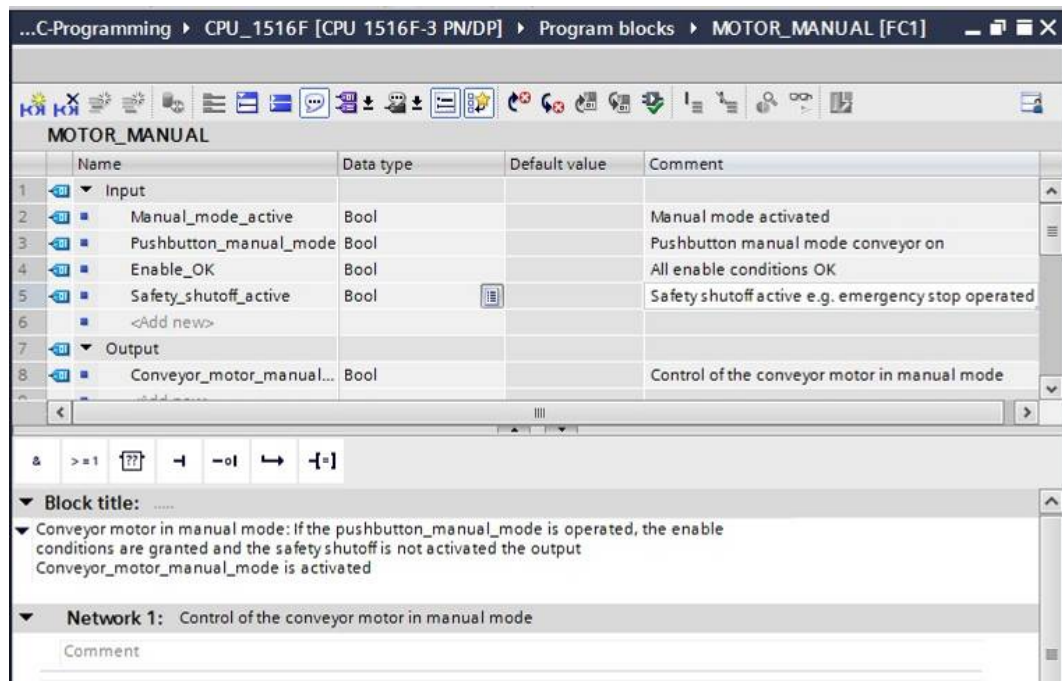
(→ Manual_mode_active → Enter → Bool → Manual mode activated)

- Add parameters #Pushbutton_manual_mode, #Enable_OK and #Safety_shutoff_active as additional binary input parameters under Input and check their data types. Add descriptive comments.



→ For purposes of program documentation, assign the block title, a block comment and a helpful network title for Network 1.

(→ Block title: Motor control in manual mode → Network 1: Control of the conveyor motor in manual mode)



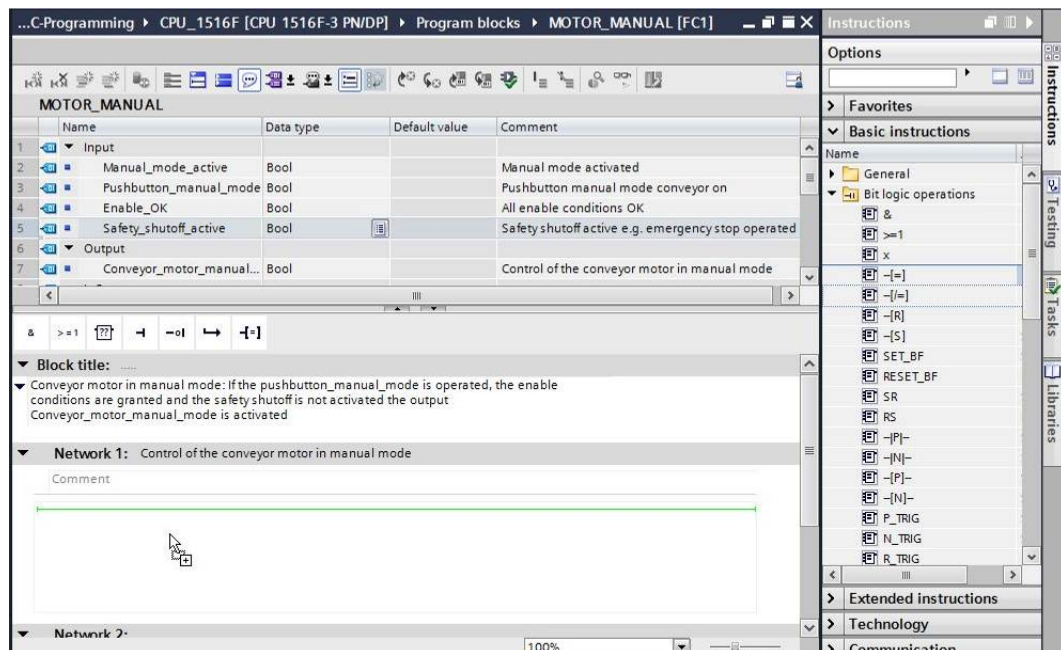
6.7 Program FC1: MOTOR_MANUAL


→ Below the interface description, you see a toolbar in the programming window with various logic functions and below that an area with networks. We have already specified the block title and the title for the first network there. Programming is performed within the networks using individual logic blocks. Distribution among multiple networks helps to preserve the clarity of the program. In the following, you will get to know the various ways you can insert logic blocks.




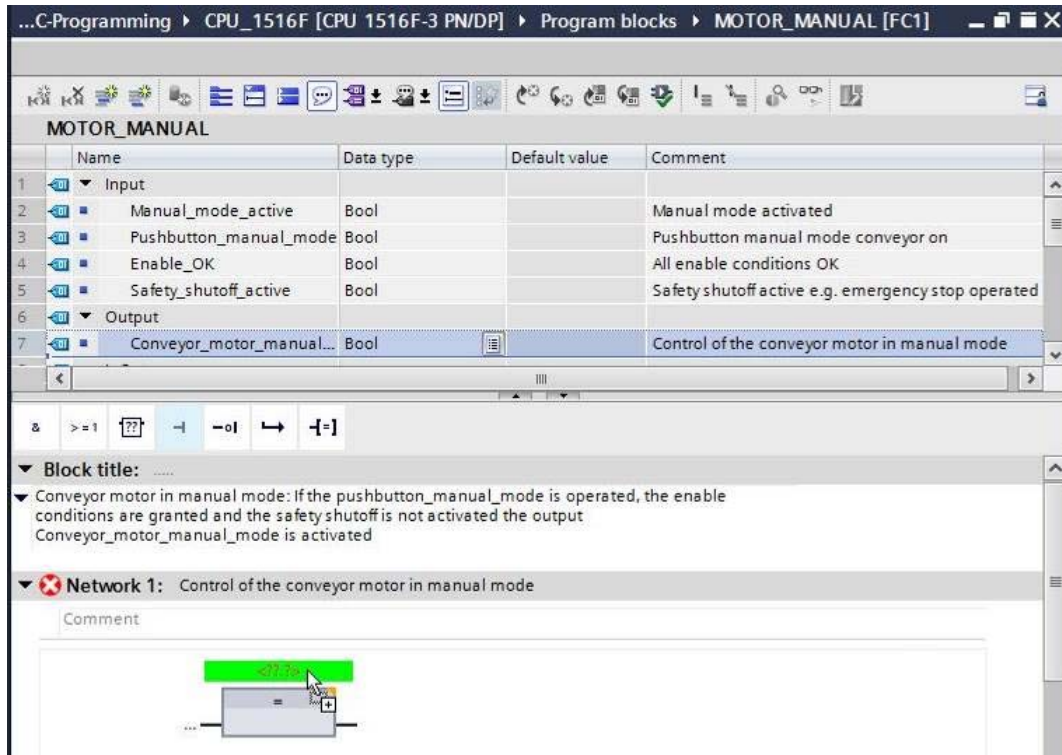
→ You can see a list of instructions you can use in the program on the right side of your programming window. Under → Basic instructions → Bit logic operations, find function –[=] (Assignment) and use a drag-and-drop operation to move it to Network 1 (green line appears, mouse pointer with + symbol).

(→ Instructions → Basic instructions → Bit logic operations → –[=])




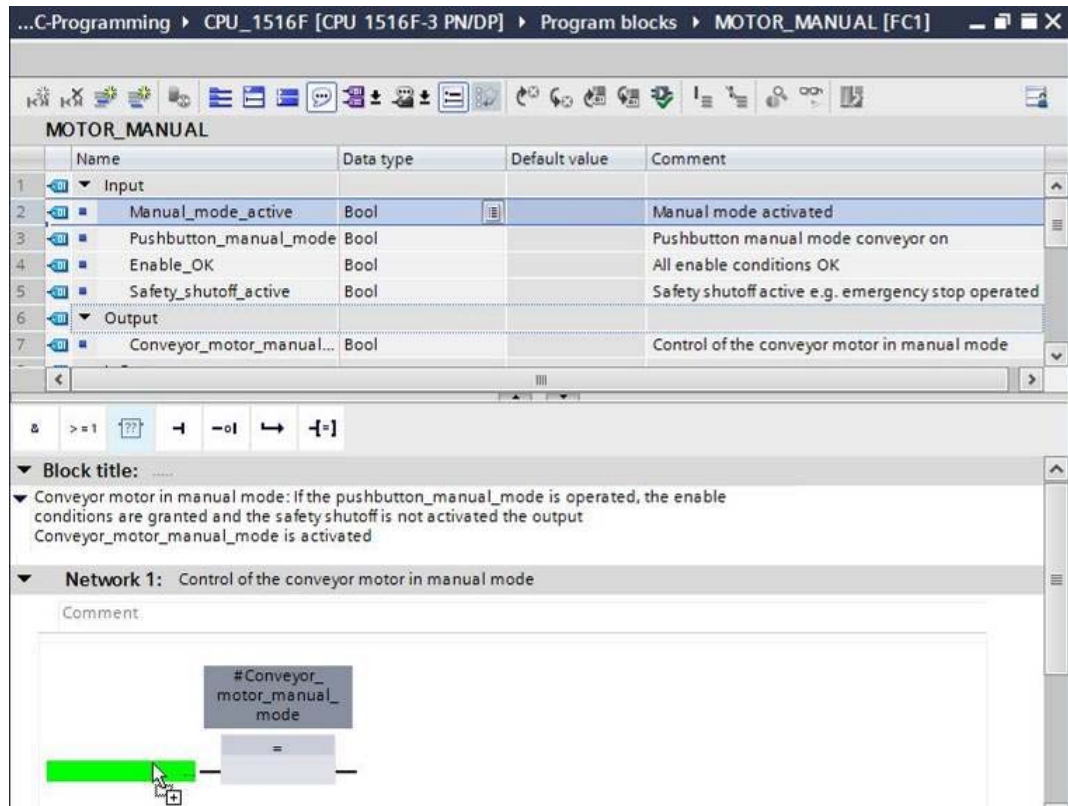
→ Now use drag-and-drop to move your output parameter #Conveyor_motor_manual_mode onto **<??.?>** above the block you just inserted. The best way to select a parameter in the interface description is by "grabbing" it at the blue symbol .

(→  Conveyor_motor_manual_mode)




- This determines that the #Conveyor_motor_manual_mode parameter is written by this block. Still missing, however, are the input conditions so that this actually happens. For this, use drag-and-drop to move input parameter #Manual_mode_active to the left side of the assignment block.

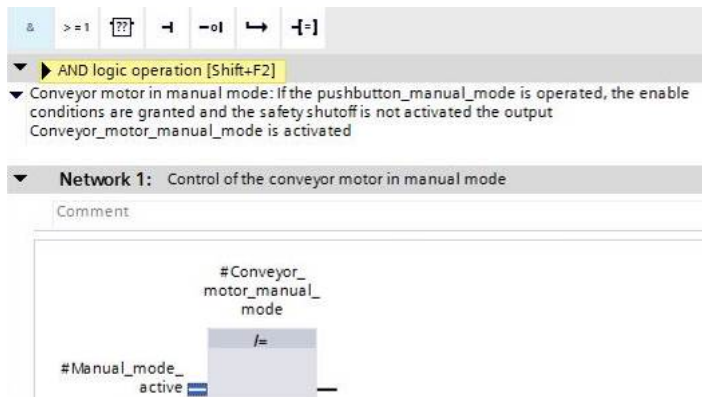
(→  Manual_mode_active)





- The input of the assignment block will also be logically combined with other parameters by an AND logic operation. To do this, first click the input of the block to which #Manual_mode_active is already connected, so that the input line has a blue background.

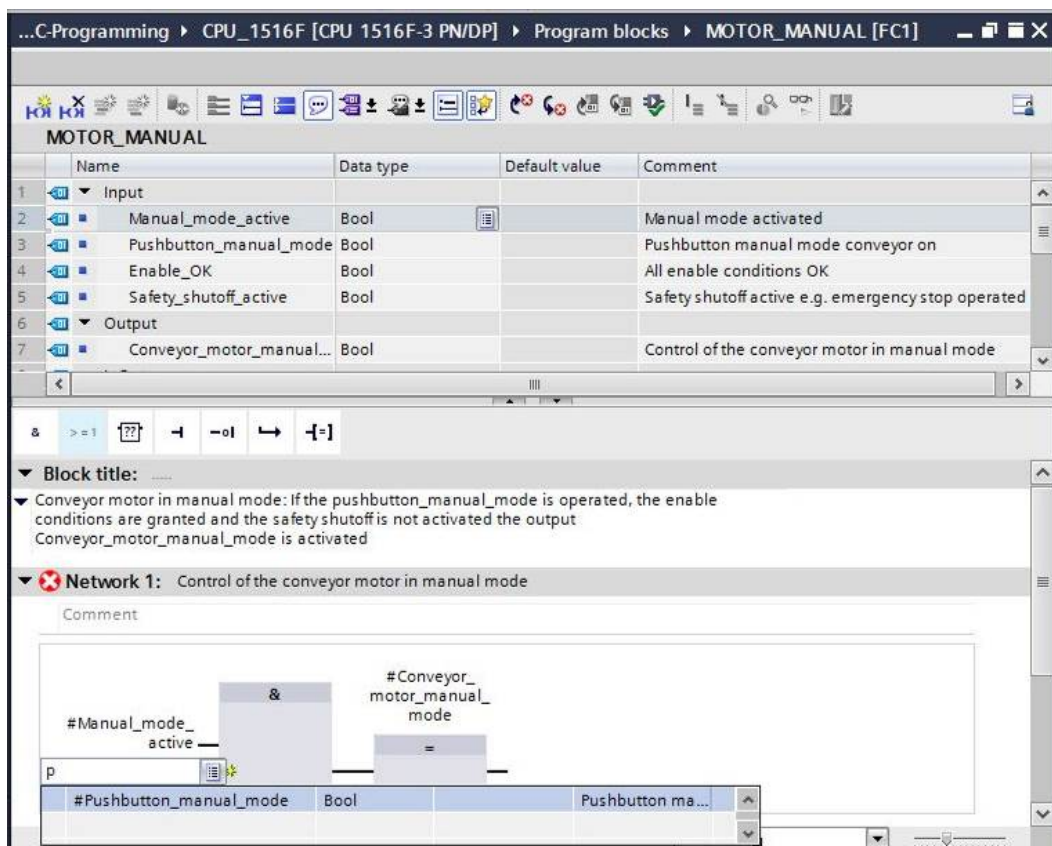


- Click the  icon in your logic toolbar to insert an AND logic operation between the #Manual_mode_active tag and your assignment block.




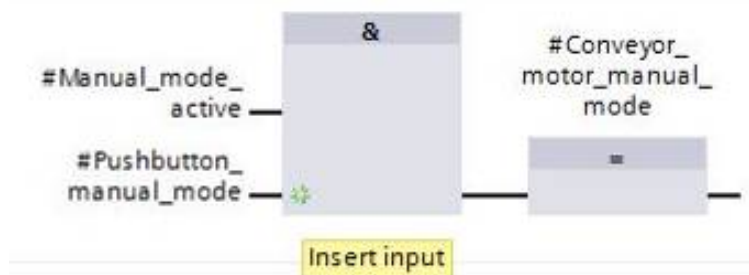
- Double-click the second input of the & logic operation  and enter the letter "P" in the field that appears in order to see a list of available tags starting with "P". Click the #Pushbutton_manual_mode tag and apply with → Enter.

(→ &- block →  → P → #Pushbutton_manual_mode → Enter)

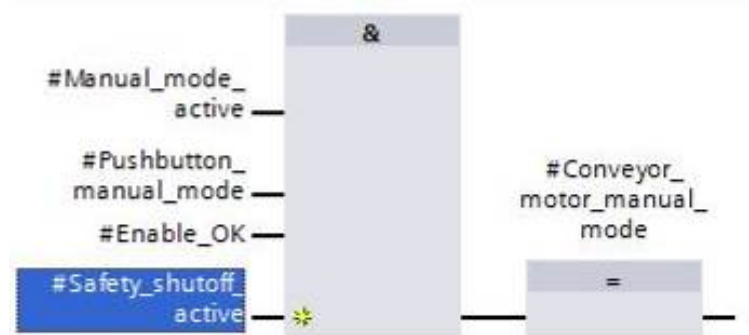


Note: When assigning tags in this way, there is a risk of a mix-up with the global tags from the tag table. The previously presented procedure using drag and drop from the interface description should therefore be used preferentially.

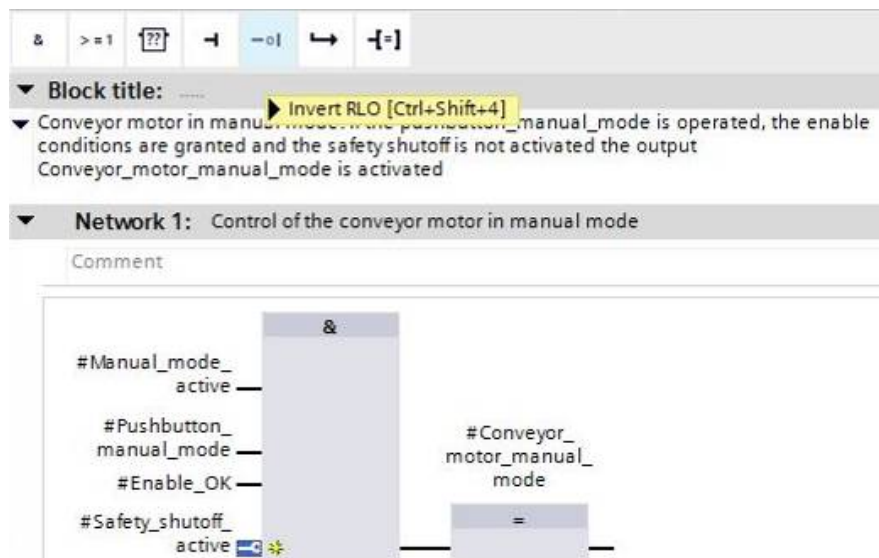
- To ensure that the output can only be controlled when the enable conditions are met and the safety shutoff is not active, the #Enable_OK and #Safety_shutoff_active input tags are logically combined with the AND logic operation. To do this, click twice on the yellow star  of your AND block to add two additional inputs.




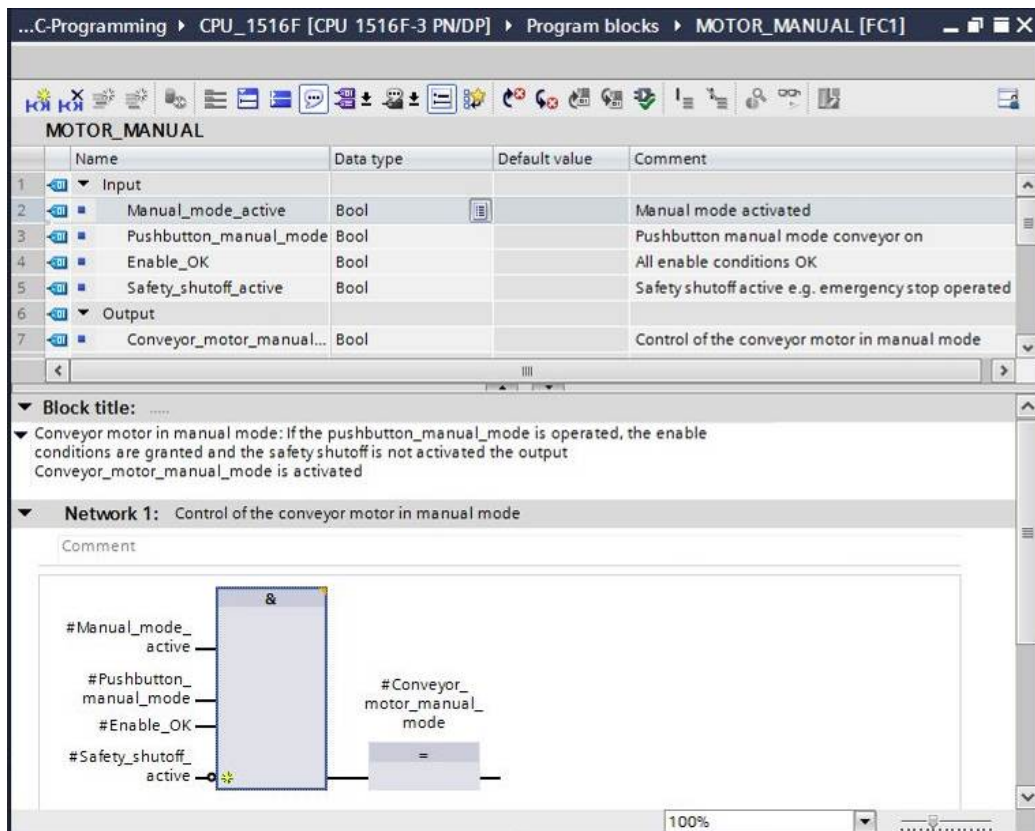
- Add input tags #Enable_OK and #Safety_shutoff_active to your newly created inputs of the AND block.



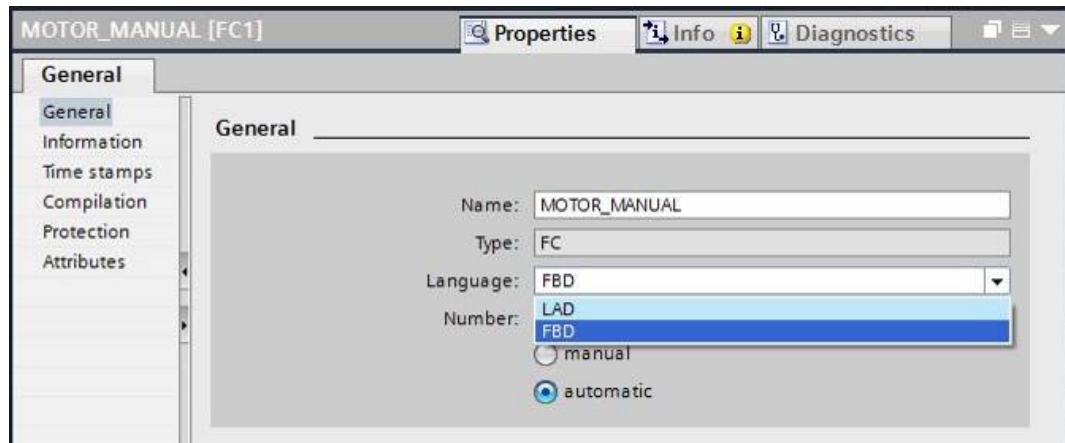
- Negate the input connected to parameter #Safety_shutoff_active by selecting it and clicking



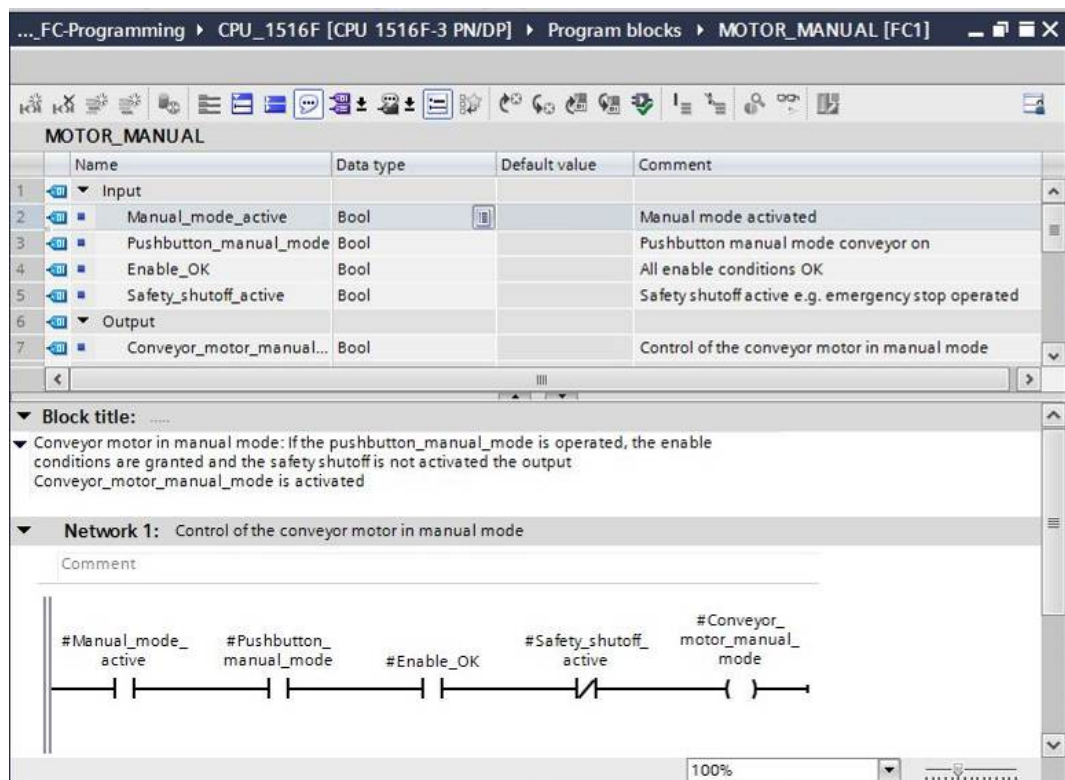
→ Do not forget to click  Save project. The finished function "MOTOR_MANUAL" [FC1] in FBD is shown below.



- Under "General" in the properties of the block, you can change the "Language" to LAD (Ladder Logic) (→Properties → General → Language: LAD)



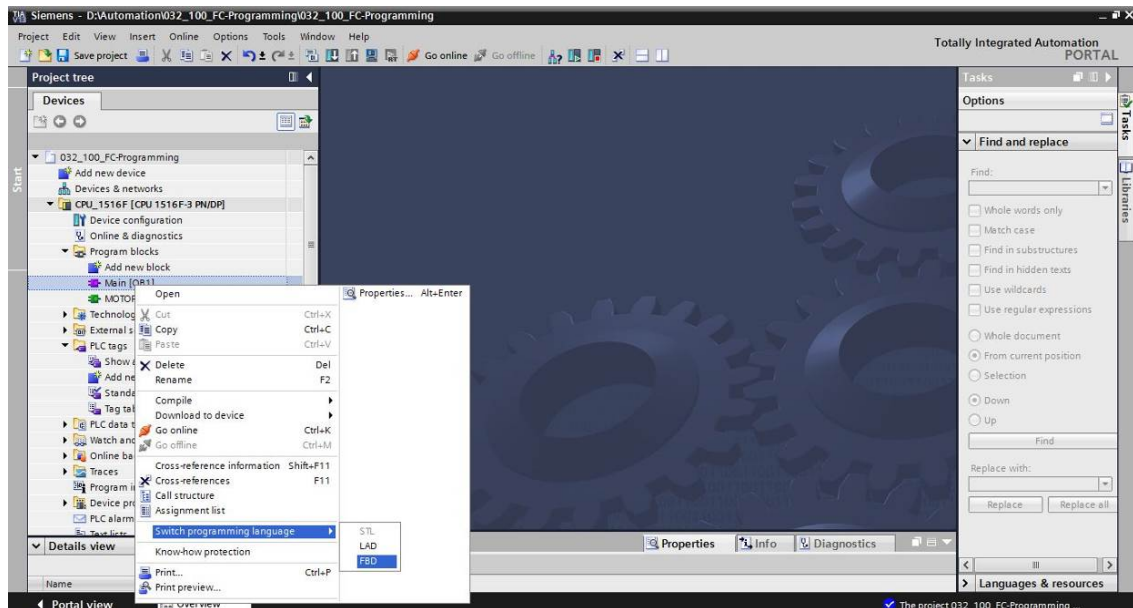
- The program has the following appearance in LAD.



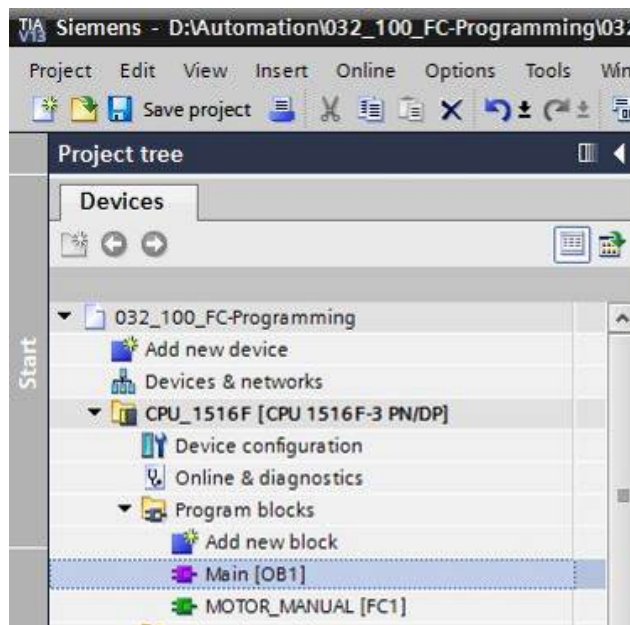
6.8 Program organization block OB1 – Control of the forward belt tracking in manual mode

→ Before programming organization block "Main [OB1]", we switch the programming language to FBD (Function Block Diagram). To do so, first click on "Main [OB1]" in the "Program blocks" folder.

(→ CPU_1516F[CPU 1516F-3 PN/DP → Program blocks → Main [OB1] → Switch programming language → FBD)

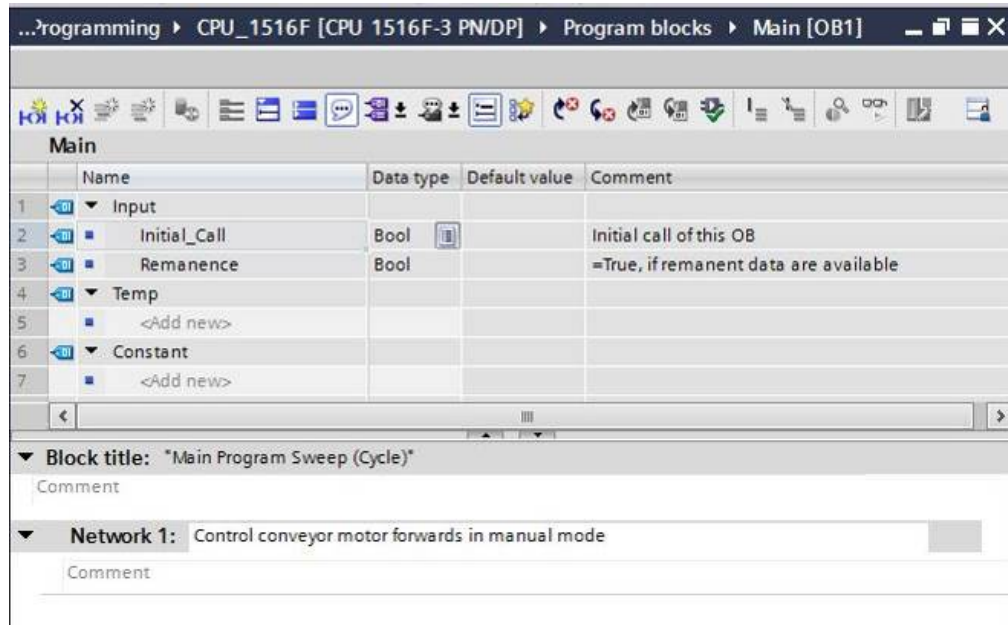


→ Open the "Main [OB1]" organization block with a double-click.

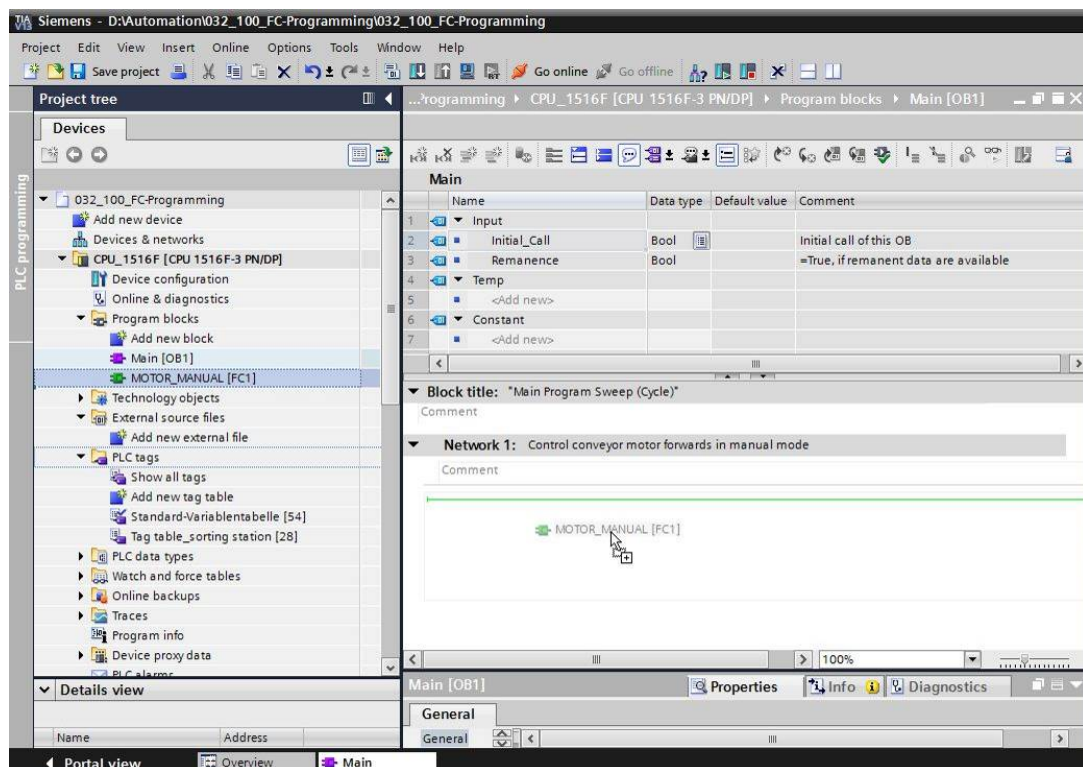


→ Assign Network 1 the name "Control conveyor tracking forward in manual/jog mode"

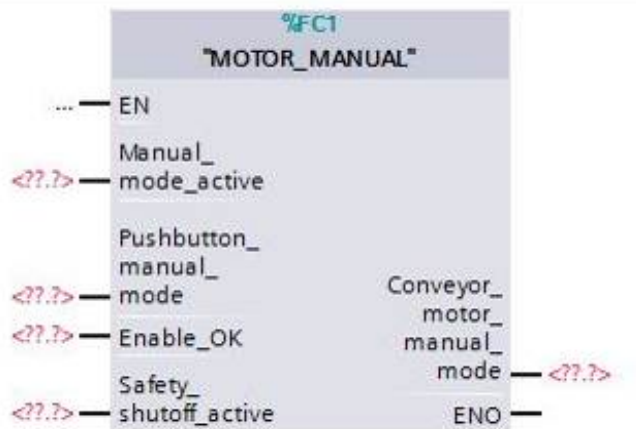
(→ Network 1:... →Control conveyor motor forwards in manual mode)





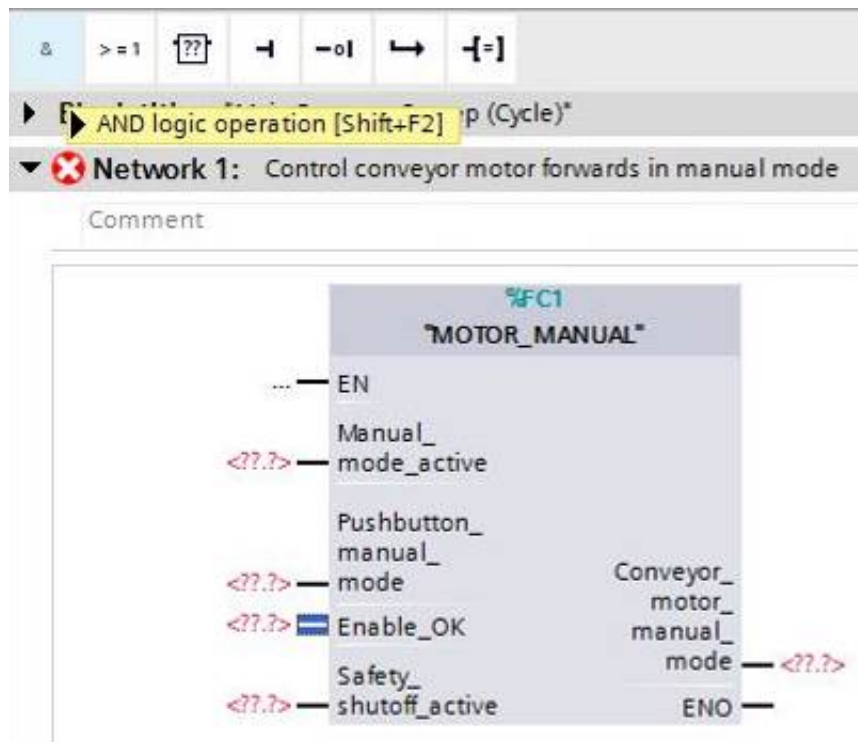
→ Use drag-and-drop to move your "MOTOR_MANUAL [FC1]" function onto the green line in Network 1.



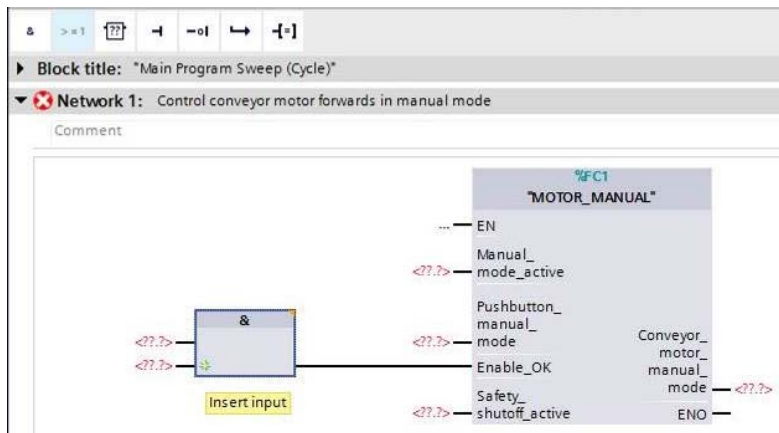
- A block with the interface you defined and connections EN and ENO are inserted in Network 1.



- To insert an AND before input parameter "Enable_OK", select this input and insert the AND by clicking the  icon in your logic toolbar (→ ).

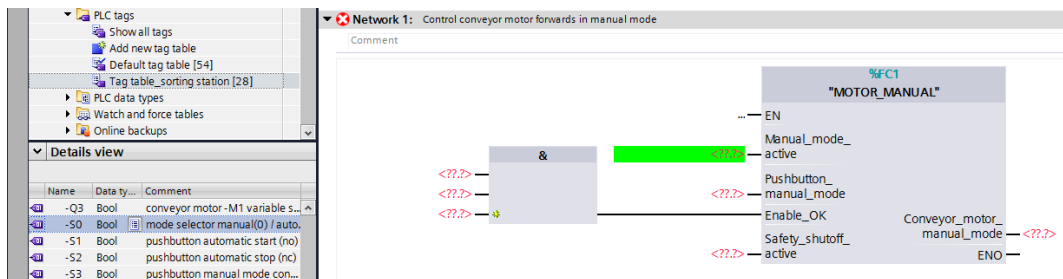


→ Click the yellow star  of the AND block to add another input (→ .

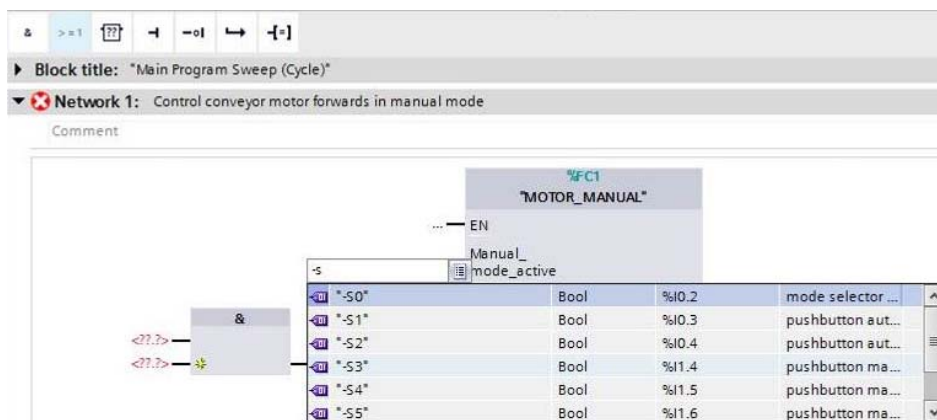


→ To connect the block to the global tags from "Tag_table_sorting_station", we have two options:

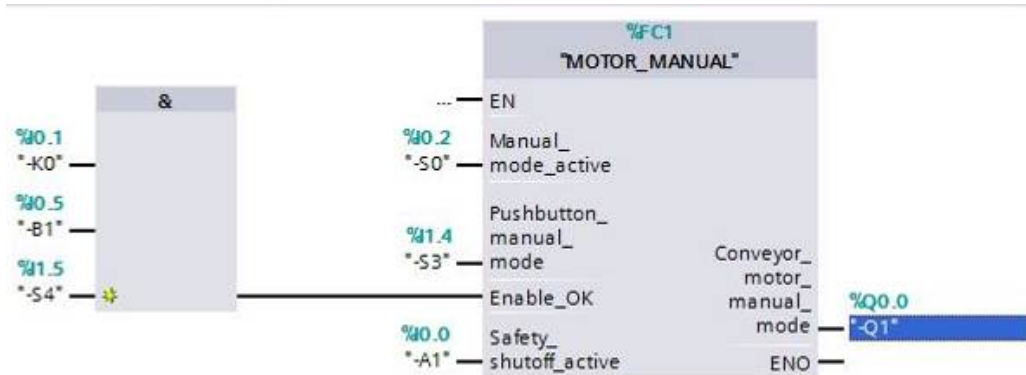
→ Either select the "Tag_table_sorting_station" in the project tree and use drag-and-drop to move the desired global tag from the Details view to the interface of FC1
(→ Tag_table_sorting_station → Details view. → -S02 → Manual_mode_active)




→ Or, enter the starting letters (e.g. "-S") of the desired global tag for <??.?> and select the global input tag "-S0" (%I0.2) from the displayed list (→ Manual_mode_active → -S → -S0).

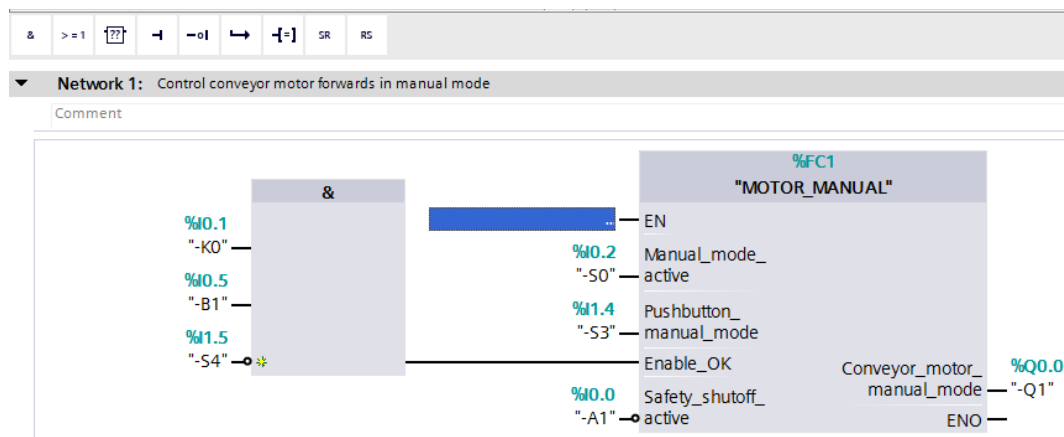


- Insert the other input tags "-S3", "-K0", "-B1", "-S4" and "-A1" and insert output tag "-Q1" (%Q0.0) at output "Conveyor_motor_manual_mode".



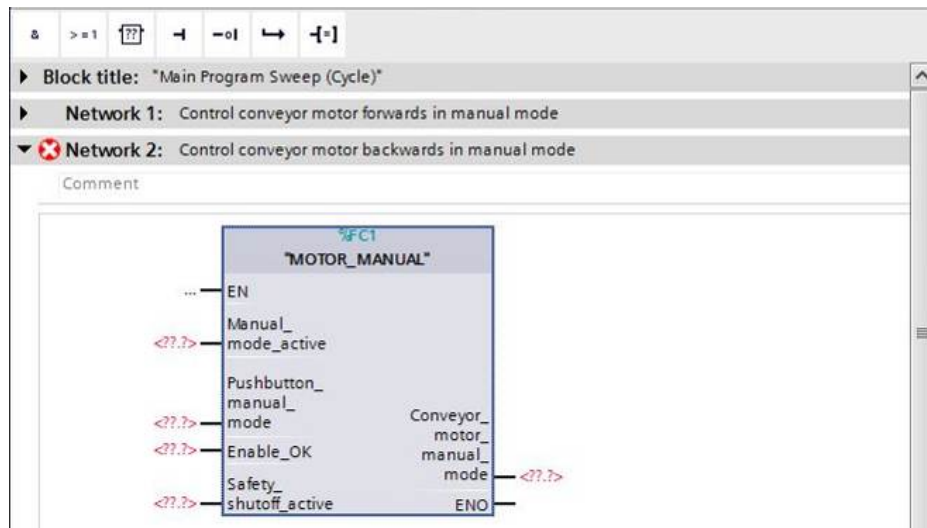
- Negate the querying of input tags "-S4" and "-A1" by selecting them and clicking 

(→ -S4 →  → -A1 → ).

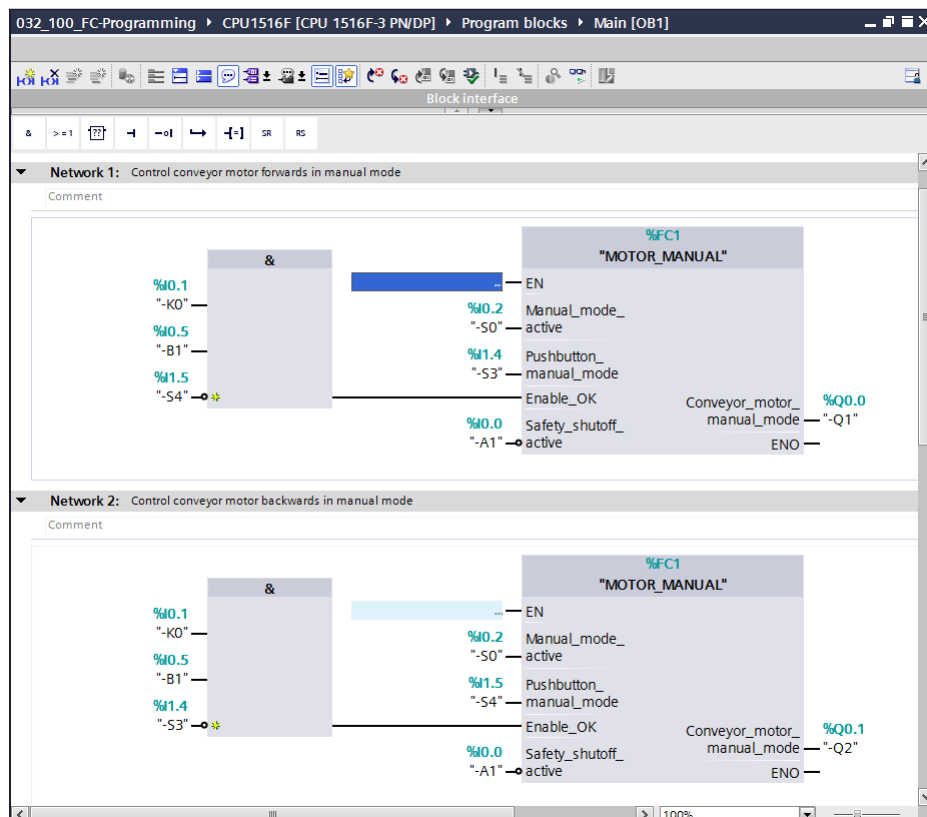


6.9 Program organization block OB1 – Control of the backward belt tracking in manual mode

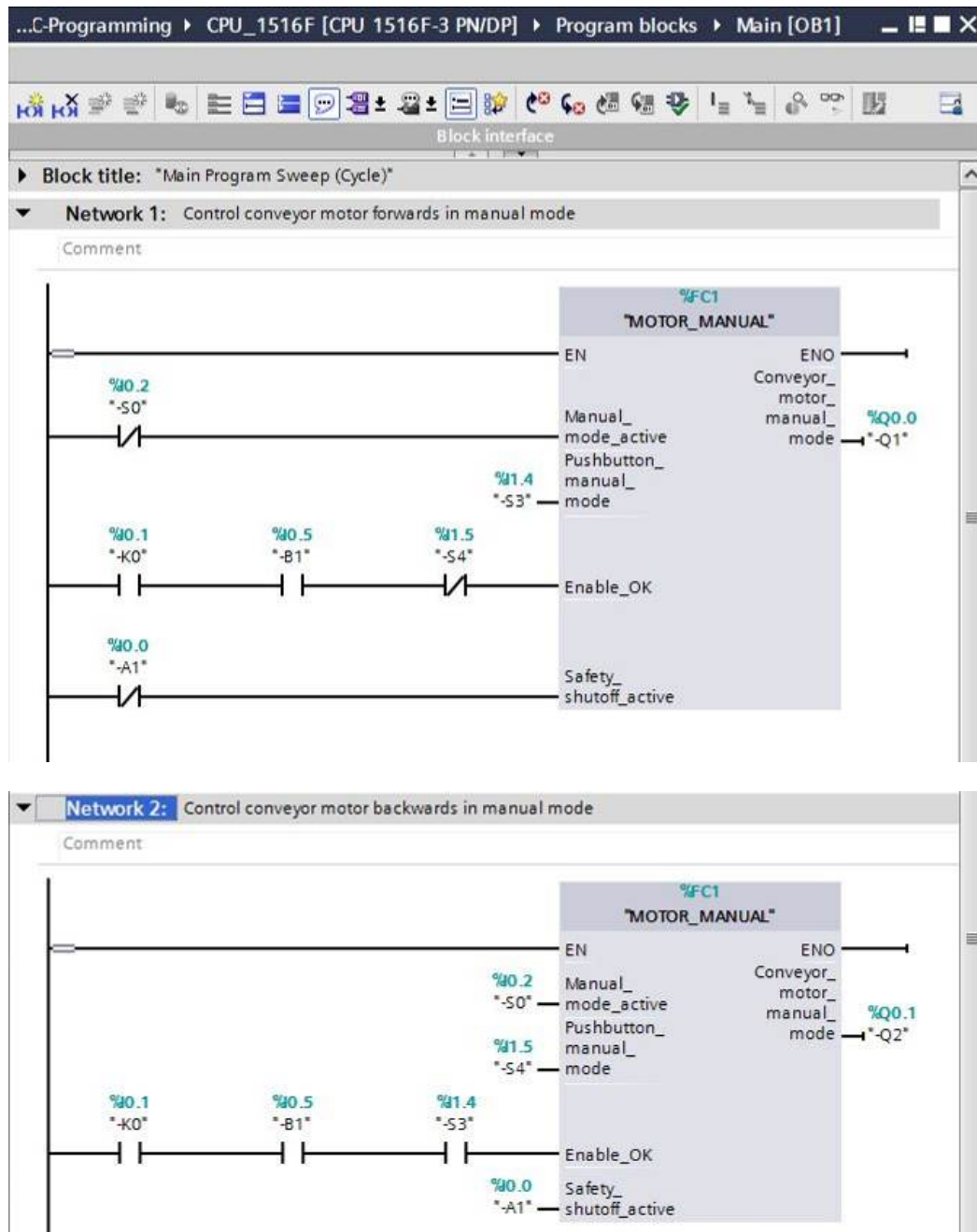
- Assign Network 2 the name "Control conveyor motor backwards in manual mode" and insert your "MOTOR_MANUAL [FC1]" function using drag-and-drop, as you did previously in Network 1.







- Connect your function as shown here. You obtain the following result in the FBD (Function Block Diagram) programming language.

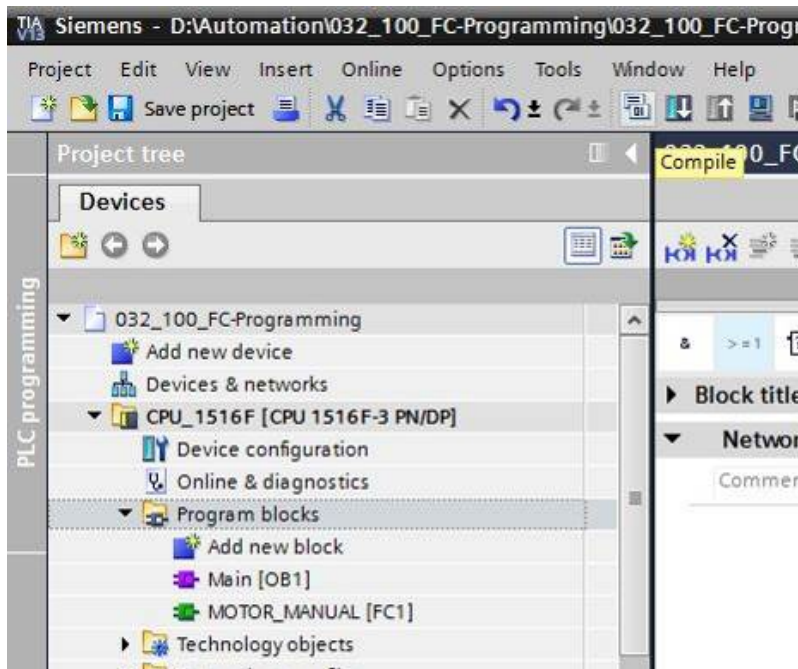


→ The result in the LAD (Ladder Logic) programming language has the following appearance.

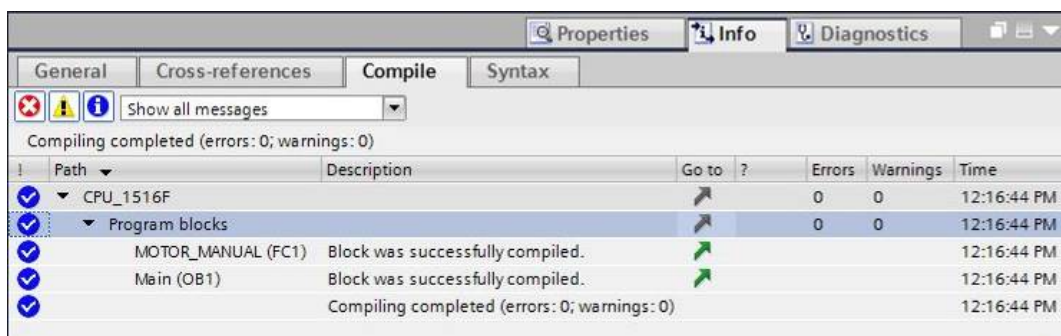


6.10 Save and compile the program

- To save your project, select the  **Save project** button in the menu. To compile all blocks, click the "Program blocks" folder and select the  icon for compiling in the menu (→  **Save project** → Program blocks → ).



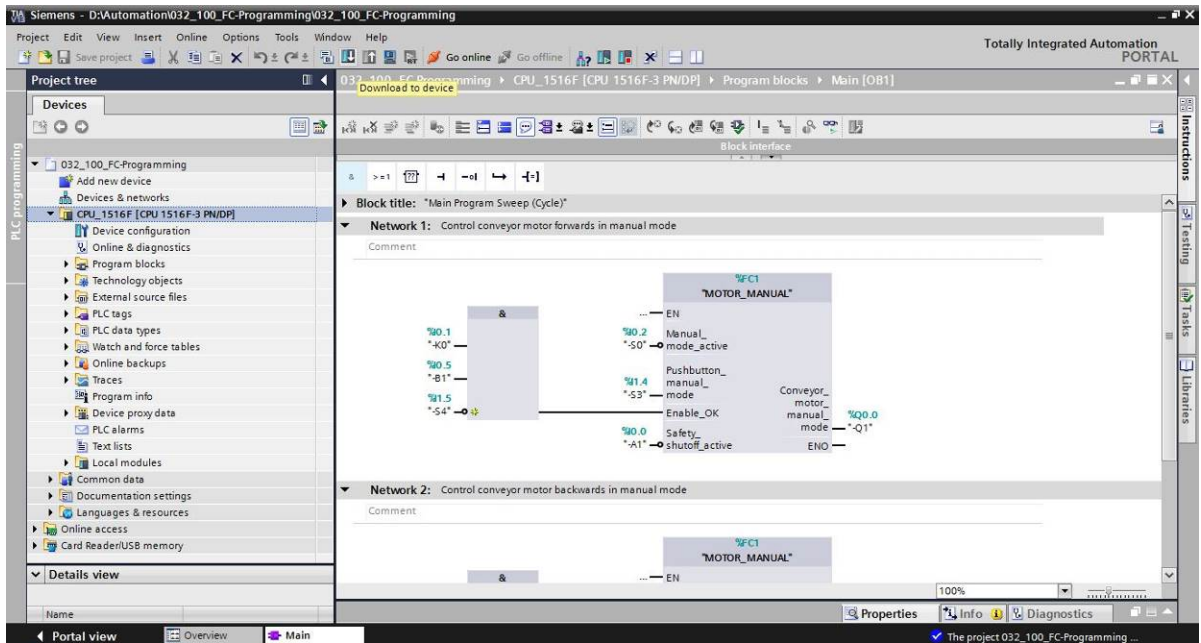
- The "Info", "Compile" area shows which blocks were successfully compiled.



6.11 Download the program

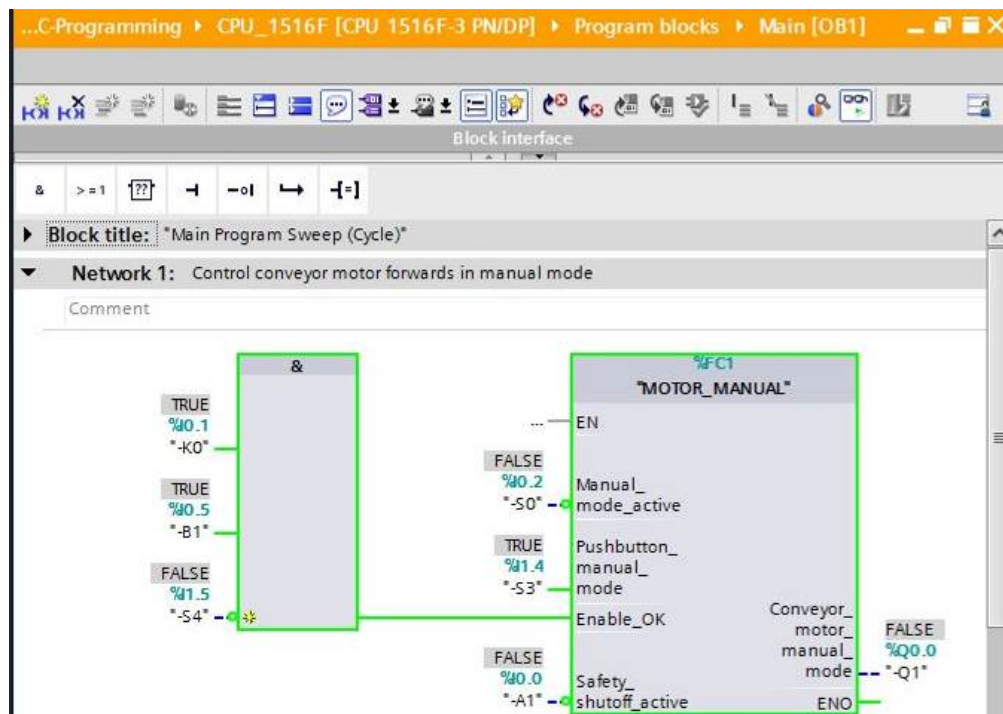
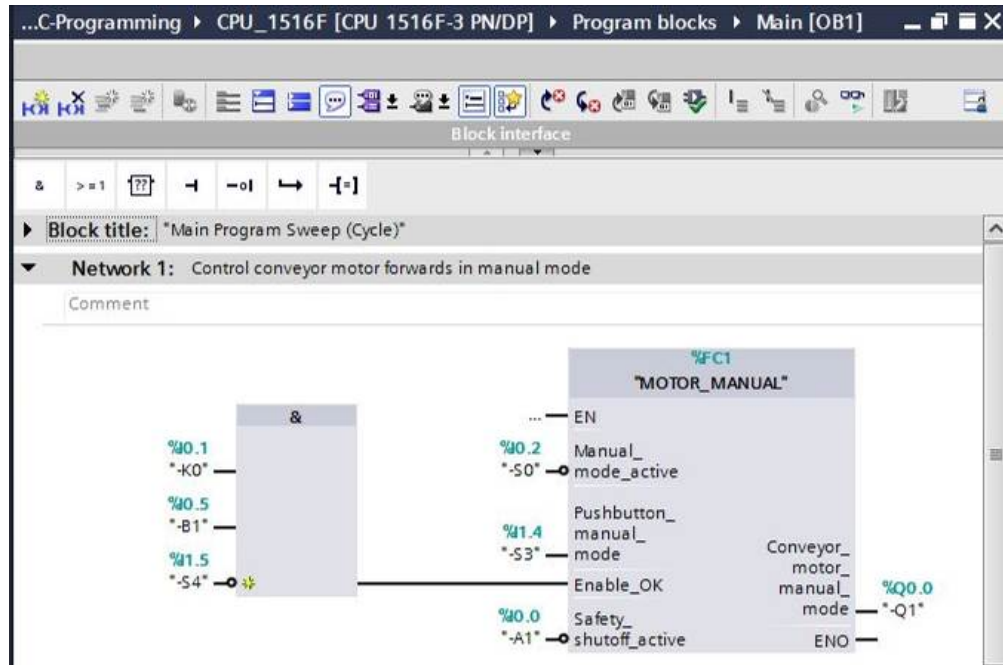
→ After successful compilation, the complete controller with the created program, as previously described in the modules for hardware configuration, can be downloaded

(→ ).



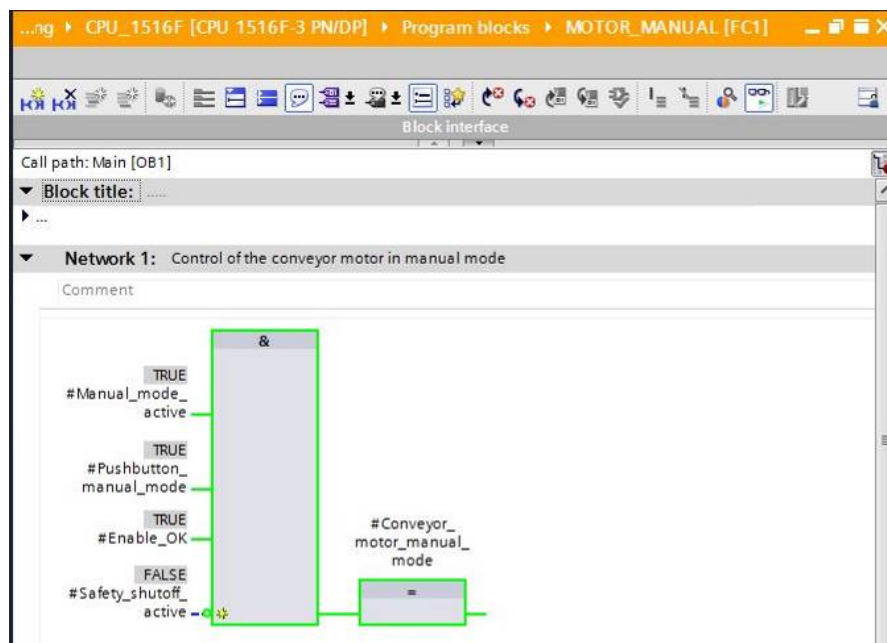
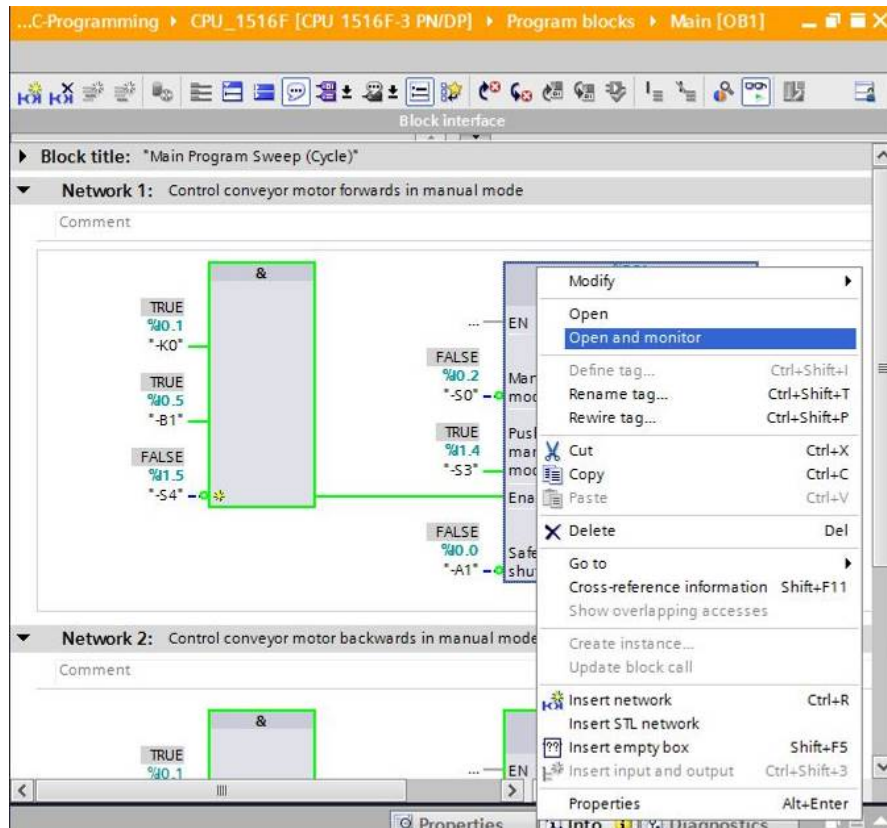
6.12 Monitor program blocks

→ The desired block must be open for monitoring the downloaded program. The monitoring can be activated/deactivated by clicking the  icon. (→ Main [OB1] → )





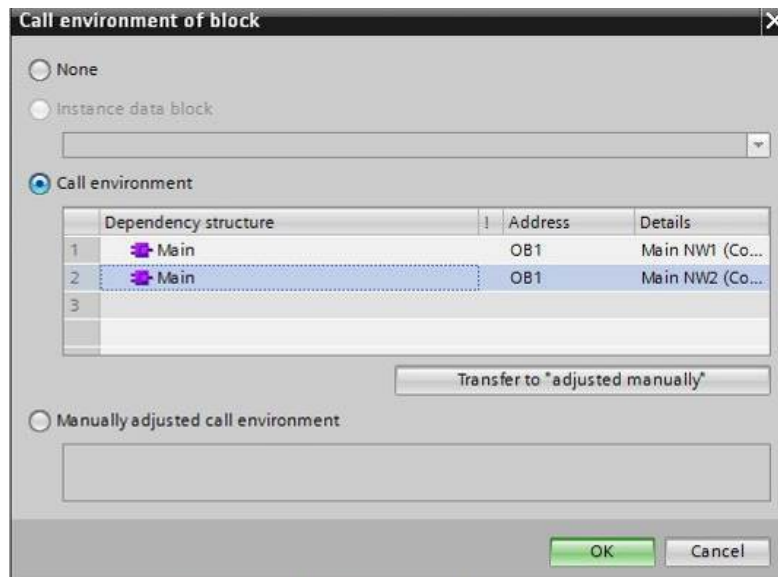
Note: The monitoring here is signal-related and controller-dependent. The signal states at the terminals are indicated with TRUE or FALSE.

- The "MOTOR_MANUAL" [FC1] function called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking (→ "MOTOR_MANUAL" [FC1] → Open and monitor).



Note: The monitoring here is function-related and controller-independent. The actuation of sensors and the station status are shown here with TRUE or FALSE.

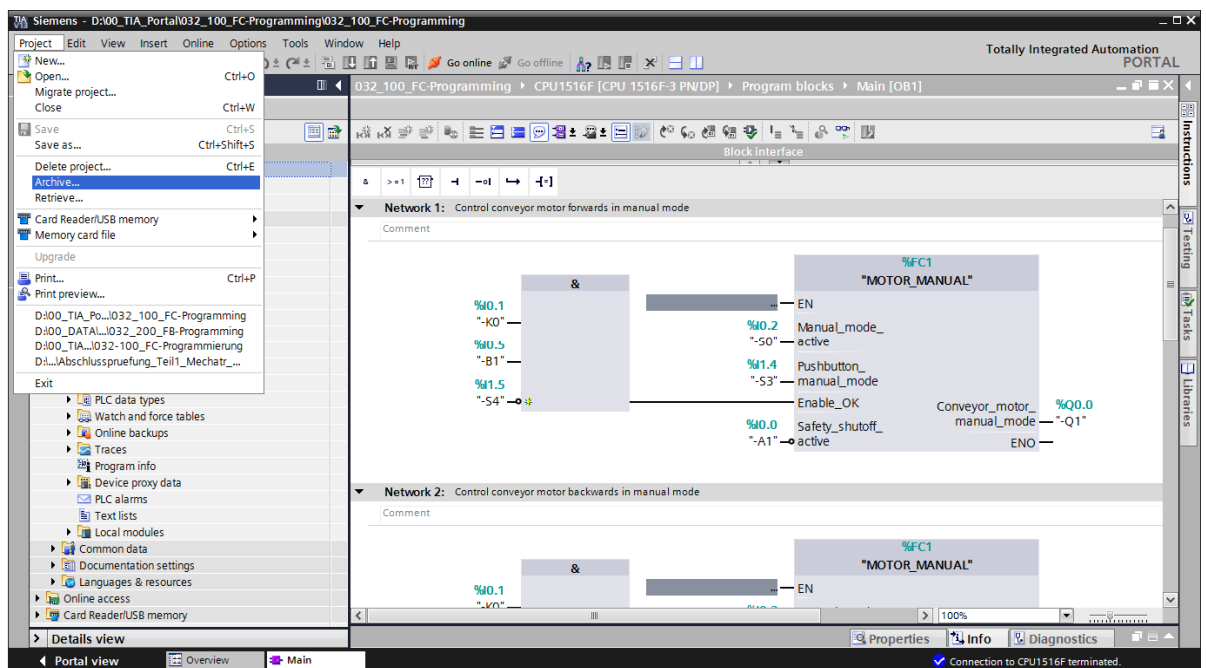
- If a particular point of use of the "MOTOR_MANUAL" [FC1] function is to be monitored, the call environment can be selected using the  icon (→  → Call environment → OK)



6

6.13 Archive the project

- As the final step, we want to archive the complete project. Select the → "Archive ..." command in the → "Project" menu. Select a folder where you want to archive your project and save it with the file type "TIA Portal project archive". (→ Project → Archive → TIA Portal project archive → 032-100_FCProgramming.... → Save)



7 Checklist

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated MANUAL mode (-S0 = 0) Activate conveyor manual mode conveyor forward (-S3 = 1) then conveyor motor forwards fixed speed (-Q1 = 1)	
4	Same as 3 but activate EMERGENCY OFF (-A1 = 0) → -Q1 = 0	
5	Same as 3 but AUTO mode (-S0 = 1) → -Q1 = 0	
6	Same as 3 but switch off station (-K0 = 0) → -Q1 = 0	
7	Same as 3 but cylinder not retracted (-B1 = 0) → -Q1 = 0	
8	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated MANUAL mode (-S0 = 0) Activate conveyor manual mode reverse (-S4 = 1) then conveyor motor backwards fixed speed (-Q2 = 1)	
9	Same as 8 but activate EMERGENCY OFF (-A1 = 0) → -Q2 = 0	
10	Same as 8 but AUTO mode (-S0 = 1) → -Q2 = 0	
11	Same as 8 but switch off station (-K0 = 0) → -Q2 = 0	
12	Same as 8 but cylinder not retracted (-B1 = 0) → -Q2 = 0	
13	Same as 8 but also activate manual mode conveyor forwards (-S3 = 1) → -Q1 = 0 and -Q2 = 0	
14	Project successfully archived	

8 Exercise

8.1 Task – Exercise

The following functions of the sorting station process description will be planned, programmed and tested in this chapter:

- Manual mode – extend cylinder
- Manual mode – retract cylinder

Note: Pay attention to the reusability or encapsulation of the functions.

8.2 Planning

Plan the implementation of the task on your own.

8.3 Checklist – Exercise

No.	Description	Completed
1	Function FC: CYLINDER_MANUAL created	
2	Interfaces defined	
3	Function programmed	
4	Function FC2 inserted in network 3 of OB1	
5	Input tags connected for Retract cylinder	
6	Output tags connected for Retract cylinder	
7	Compiling successful and without error message	
8	Function FC2 inserted in network 4 of OB1	
9	Input tags connected for Extend cylinder	
10	Output tags connected for Extend cylinder	
11	Compiling successful and without error message	
12	Download successful and without error message	
13	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated MANUAL mode (-S0 = 0) Do not activate Retract cylinder (-S5 = 0) Activate Extend cylinder (-S6 = 1) then extend cylinder (-M3 = 1) successfully	
14	Switch on station (-K0 = 1) Cylinder extended / Feedback activated (-B2 = 0) EMERGENCY OFF (-A1 = 1) not activated MANUAL mode (-S0 = 0) Do not activate Extend cylinder (-S6 = 0) Activate Retract cylinder (-S5 = 1) then retract cylinder (-M2 = 1) successful	
15	Retract cylinder and Extend cylinder cannot be activated simultaneously	
16	Project successfully archived	

9 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

6

[illegible]

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, get in touch with your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	4
2	Prerequisite.....	4
3	Theory.....	4
3.1	Operating system and application program	4
3.2	Organization blocks.....	5
3.3	Process image and cyclic program processing.....	6
3.4	Functions.....	8
3.5	Function blocks and instance data blocks.....	9
3.6	Global data blocks.....	10
3.7	Library-compatible code blocks.....	11
3.8	Programming languages	12
4	Task	13
5	Planning.....	13
5.1	EMERGENCY STOP	13
5.2	Automatic mode - Conveyor motor	13
6	Structured step-by-step instructions	14
6.1	Retrieve an existing project.....	14
6.2	Create a new tag table	15
6.3	Create new tags within a tag table	17
6.4	Import "Tag_table_sorting_station"	18
6.5	Create function block FB1 "MOTOR_AUTO" for the conveyor motor in automatic mode	22
6.6	Define the interface of FB1 "MOTOR_AUTO"	24
6.7	Program FB1: MOTOR_AUTO	27
6.8	Program organization block OB1 – Control of the forward belt tracking in automatic mode	35
6.9	The result in the LAD (Ladder Logic) programming language has the following appearance....	40
6.10	Save and compile the program	41
6.11	Download the program	42
6.12	Monitor program blocks.....	43
6.13	Archive the project.....	46
7	Checklist	47
8	Exercise	48
8.1	Task – Exercise.....	48
8.2	Planning	48
8.3	Checklist – Exercise	49
9	Additional information	50

BASICS OF FB PROGRAMMING

1 Goal

In this chapter, you will get to know the basic elements of a control program – the **organization blocks (OBs)**, **functions (FCs)**, **function blocks (FBs)** and **data blocks (DBs)**. In addition, you will be introduced to **library-compatible** function and function block programming. You will get to know the **Function Block Diagram (FBD)** programming language and use it to program a function block (FB1) and an organization block (OB1).

2 Prerequisite

This chapter builds on the hardware configuration of SIMATIC S7 CPU1516F-3 PN/DP. However, other hardware configurations that have digital input and output cards can be used. You can use the following project for this chapter, for example:

SCE_EN_012_101__Hardware_Configuration_CPU1516F.zap13

7

3 Theory

3.1 Operating system and application program

Every controller (CPU) contains an **operating system**, which organizes all functions and processes of the CPU that are not associated with a specific control task. The tasks of the operating system include the following:

- Performing a warm restart
- Updating the process image of the inputs and outputs
- Cyclically calling the user program
- Detecting interrupts and calling interrupt OBs
- Detecting and handling errors
- Managing memory areas

The operating system is an integral component of the CPU and comes pre-installed.

The **user program** contains all functions that are necessary for executing your specific automation task. The tasks of the user program include the following:

- Checking the basic requirements for a warm restart using startup OBs
- Processing of process data, i.e. activation of output signals as a function of the input signal states
- Reaction to interrupts and interrupt inputs
- Error handling during normal program execution

3.2 Organization blocks

Organization blocks (OBs) form the interface between the operating system of the controller (CPU) and the application program. They are called from the operating system and control the following operations:

- Cyclic program processing (e.g. OB1)
- Startup characteristics of the controller
- Interrupt-driven program processing
- Error handling

A project must have **an organization block for cyclic program processing** at a minimum. An OB is called by a **start event** as shown in Figure 1. In addition, the individual OBs have defined priorities so that, for example, an OB82 for error handling can interrupt the cyclic OB1.

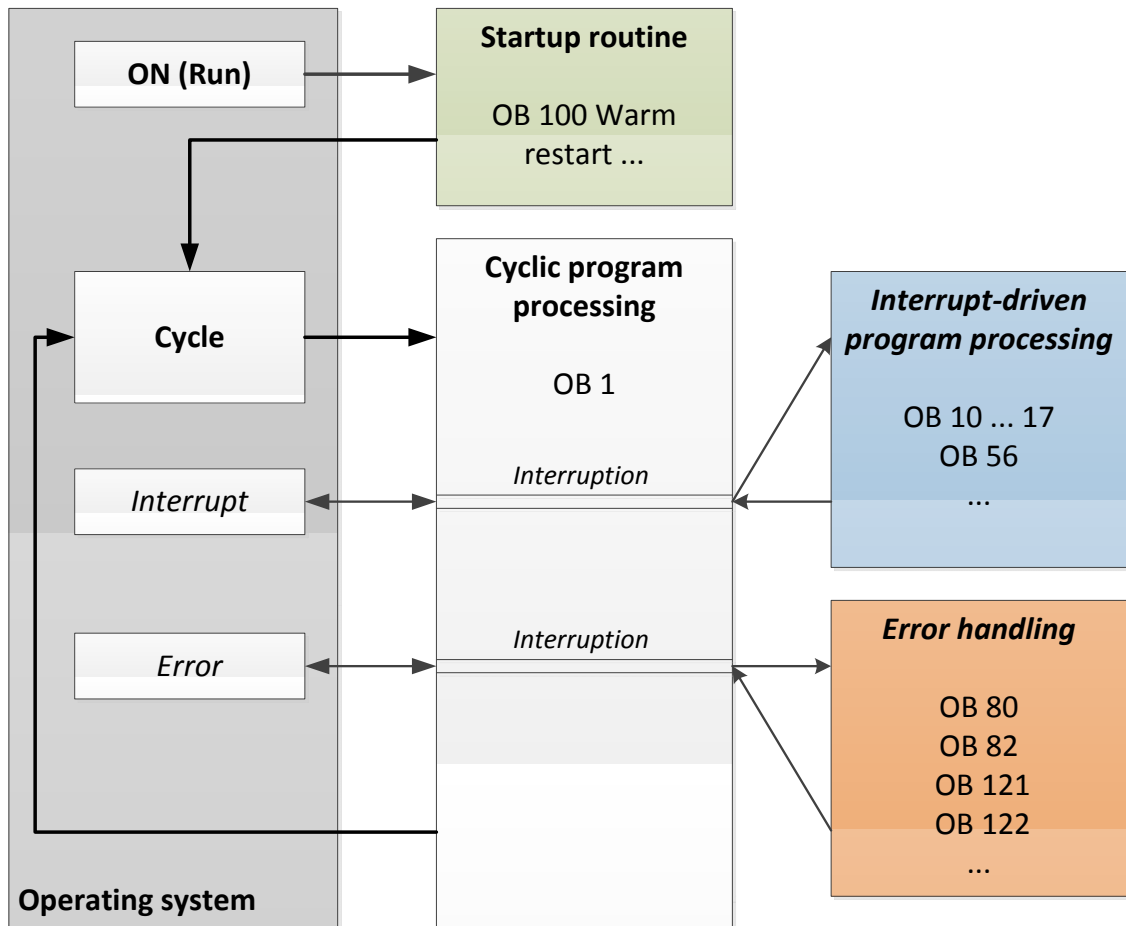


Figure 1: Start events in the operating system and OB calls

When a start event occurs, the following reactions are possible:

- If an OB has been assigned to the event, this event triggers the execution of the assigned OB. If the priority of the assigned OB is greater than the priority of the OB that is currently being executed, it is executed immediately (interrupt). If not, the assigned OB waits until the higher-priority OB has been completely executed.
- If an OB is not assigned to the event, the default system reaction is performed.

Table 1 gives a couple of examples of start events for a SIMATIC S7-1500, their possible OB number(s) and the default system reaction in the event the organization block is not present in the controller.

Start event	Possible OB numbers	Default system reaction
Startup	100, ≥ 123	Ignore
Cyclic program	1, ≥ 123	Ignore
Time-of-day interrupt	10 to 17, ≥ 123	-
Update interrupt	56	Ignore
Scan cycle monitoring time exceeded once	80	STOP
Diagnostic interrupt	82	Ignore
Programming error	121	STOP
I/O access error	122	Ignore

Table 1: OB numbers for various start events

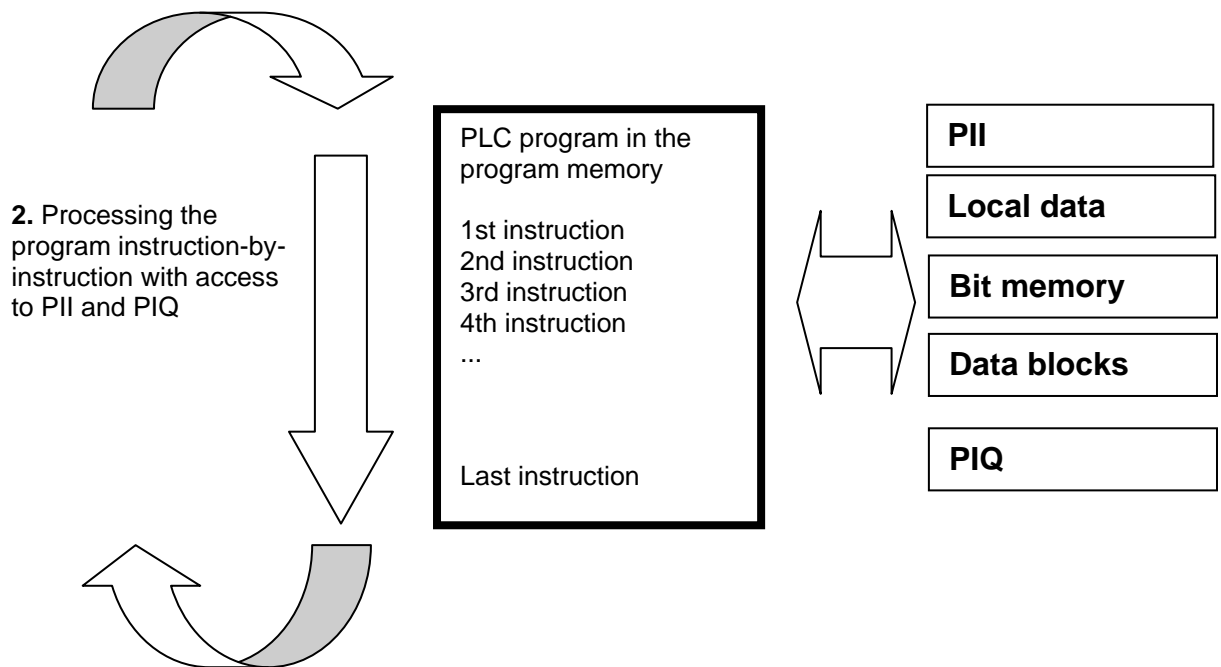
3.3 Process image and cyclic program processing

When the cyclic user program addresses the inputs (I) and outputs (O), it does not query the signal states directly from the input/output modules. Instead, it accesses a memory area of the CPU. This memory area contains an image of the signal states and is called the **process image**.

The cyclic program processing sequence is as follows:

1. At the start of the cyclic program, a query is sent to determine whether or not the individual inputs are energized. This status of the inputs is stored in the **process image of the inputs (PII)**. In doing so, the information 1 or "High" is stored for energized inputs and the information 0 or "Low" for de-energized inputs.
2. The CPU now executes the program stored in the cyclic organization block. For the required input information, the CPU accesses the previously read **process image of the inputs (PII)** and the results of logic operation (RLOs) are written to a so-called **process image of the outputs (PIQ)**.
3. At the end of the cycle, the **process image of the outputs (PIQ)** is transferred as the signal state to the output modules and these are energized or de-energized. The sequence then continues again with Item 1.

1. Save status of inputs in PII.



3. Transfer status from the PIQ to the outputs.

Figure 2: Cyclic program processing

Note: The time the CPU needs for this sequence is called cycle time. This depends, in turn, on the number and type of instructions and the processor performance of the controller.

3.4 Functions

Functions (FCs) are logic blocks without memory. They **have no data memory** in which values of block parameters can be stored. Therefore, all interface parameters must be connected when a function is called. To store data permanently, global data blocks must be created beforehand.

A function contains a program that is executed whenever the function is called from another code block.

Functions can be used, for example, for the following purposes:

- Math functions – that return a result dependent on input values.
- Technological functions – such as individual controls with binary logic operations.

A function can also be called several times at different points within a program.

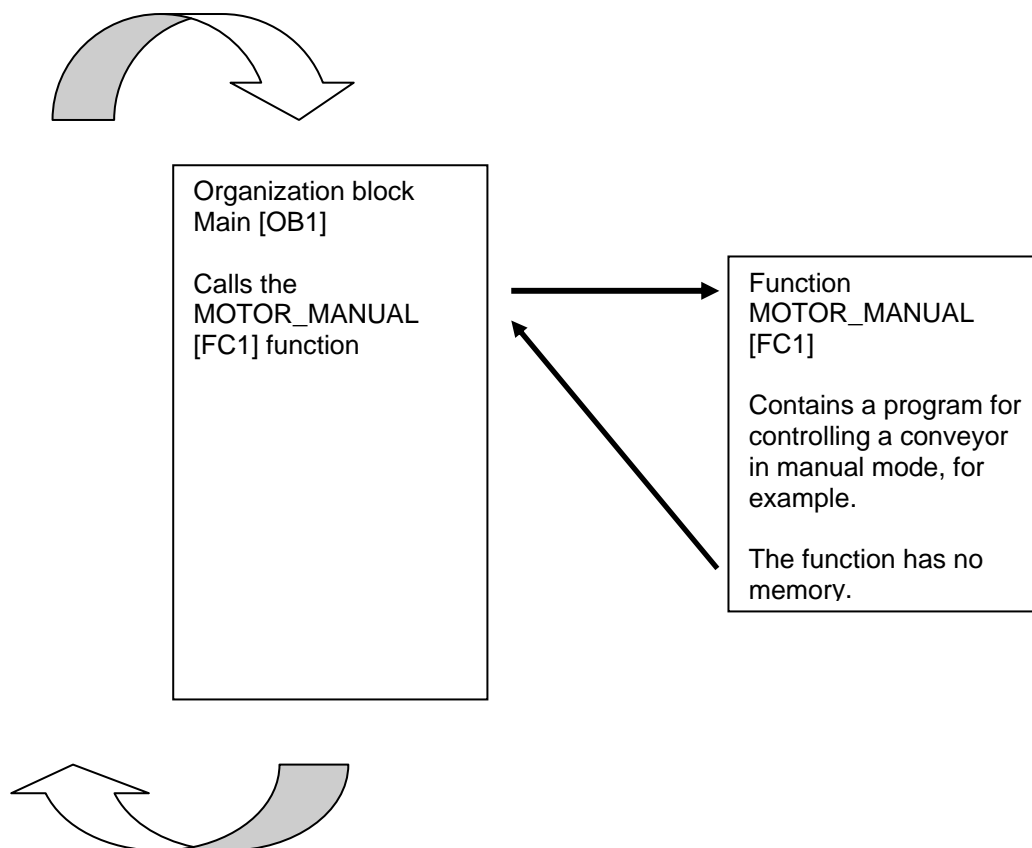


Figure 3: Function with call from organization block Main [OB1]

3.5 Function blocks and instance data blocks

Function blocks are code blocks that store their input, output and in-out tags as well as static tags permanently in instance data blocks, so that they **are available after the block has been executed**. For this reason, they are also referred to as blocks with "memory".

Function blocks can also operate with temporary tags. Temporary tags are not stored in the instance DB, however. Instead, they are only available for one cycle.

Function blocks are used for tasks that cannot be implemented with functions:

- Whenever timers and counters are required in the blocks.
- Whenever information must be saved in the program, such as pre-selection of the operating mode with a button.

Function blocks are always executed when called from another code block. A function block can also be called several times at different points within a program. This facilitates the programming of frequently recurring complex functions.

A call of a function block is referred to as an instance. Each instance of a function block is assigned a memory area that contains the data that the function block uses. This memory is made available by data blocks created automatically by the software.

It is also possible to provide memory for multiple instances in one data block in the form of a **multi-instance**. The maximum size of instance data blocks varies depending on the CPU. The tags declared in the function block determine the structure of the instance data block.

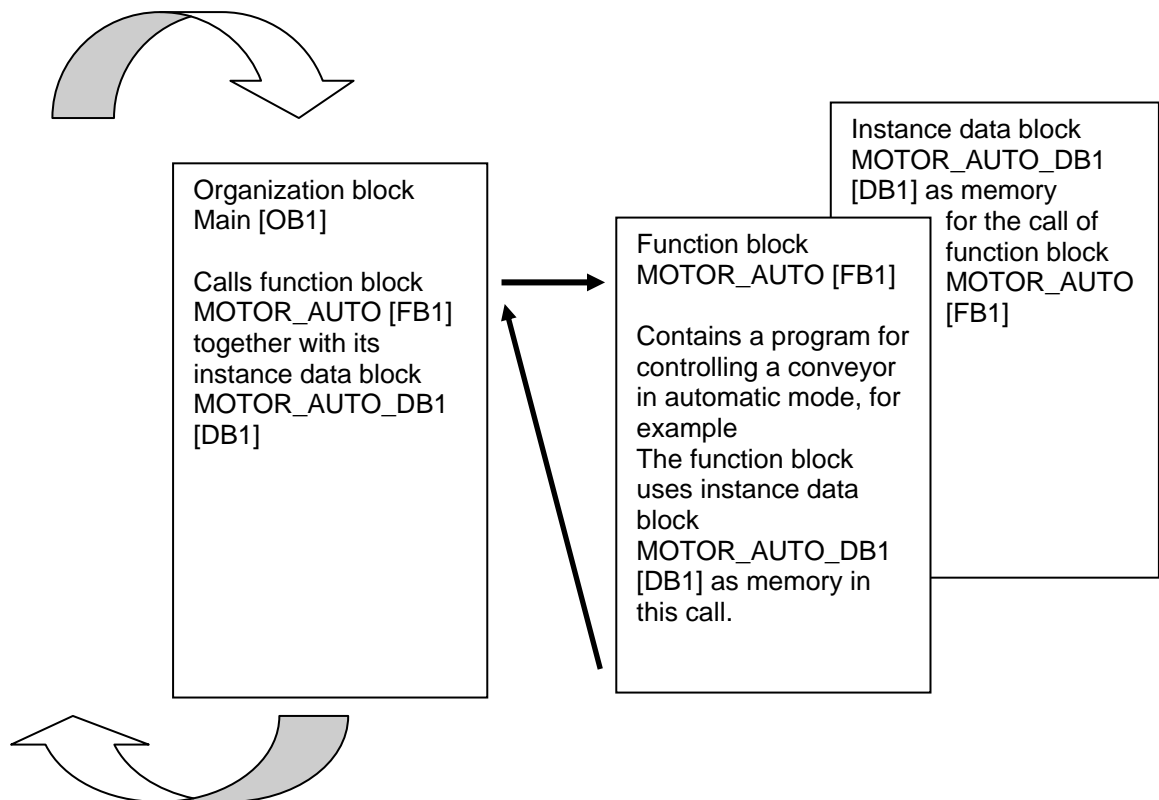


Figure 4: Function block and instance with call from organization block Main [OB1]

3.6 Global data blocks

In contrast to logic blocks, data blocks contain no instructions. Rather, they serve as memory for user data.

Data blocks thus contain variable data that is used by the user program. You can define the structure of global data blocks as required.

Global data blocks store data that can be used **by all other blocks** (see Figure 5). Only the associated function block should access instance data blocks. The maximum size of data blocks varies depending on the CPU.

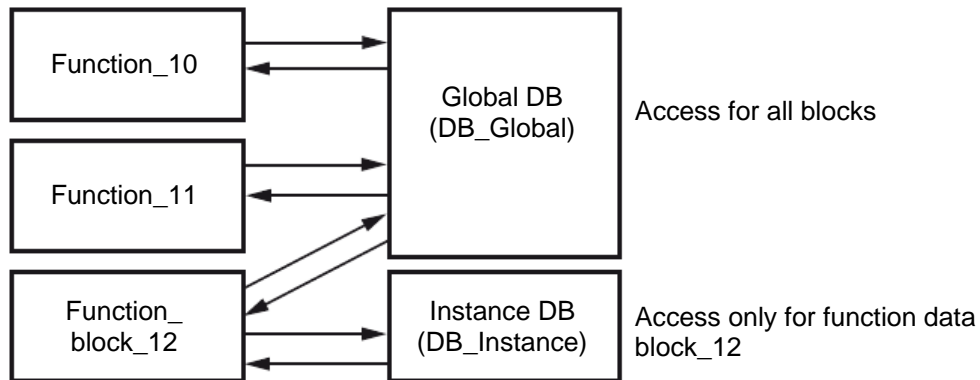


Figure 5: Difference between global DB and instance DB.

Application examples for **global data blocks** are:

- Saving of information about a storage system. "Which product is located where?"
- Saving of recipes for particular products.

3.7 Library-compatible code blocks

A user program can be created with linear or structured programming. **Linear programming** writes the entire user program in the cycle OB, but is only suitable for very simple programs for which other less expensive control systems, such as LOGO!, can now be used.

Structured programming is always recommended for more complex programs. Here, the overall automation task can be broken down into small sub-tasks in order to implement a solution for them in functions and function blocks.

In this case, library-compatible logic blocks should be created preferentially. This means that the input and output parameters of a function or function block are defined generally and only supplied with the current global tags (inputs/outputs) when the block is used.

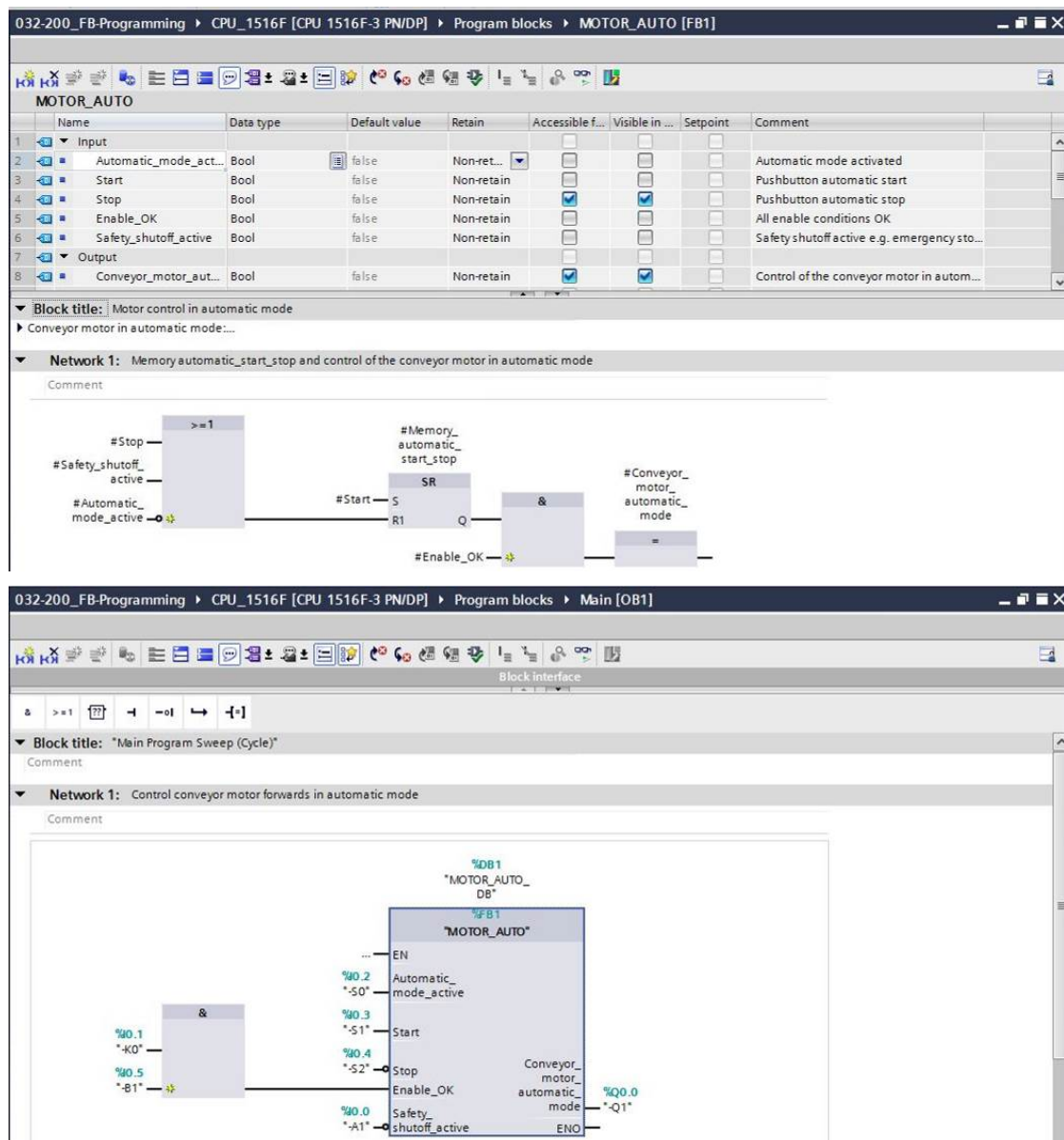


Figure 6: Library-compatible function with call in OB1

3.8 Programming languages

The available programming languages for programming functions are Function Block Diagram (FBD), Ladder Logic (LAD), Statement List (STL) and Structured Control Language (SCL). For function blocks, the GRAPH programming language is additionally available for programming graphical step sequences.

The **Function Block Diagram (FBD)** programming language will be presented in the following.

FBD is a graphical programming language. The representation is based on electronic switching systems. The program is mapped in networks. A network contains one or more logic operation paths. Binary and analog signals are linked by boxes. The graphical logic symbols known from Boolean algebra are used to represent the binary logic.

You can use binary functions to query binary operands and to logically combine their signal states. The following instructions are examples of binary functions: "AND operation", "OR operation" and "EXCLUSIVE OR operation". These are shown in Figure 7.

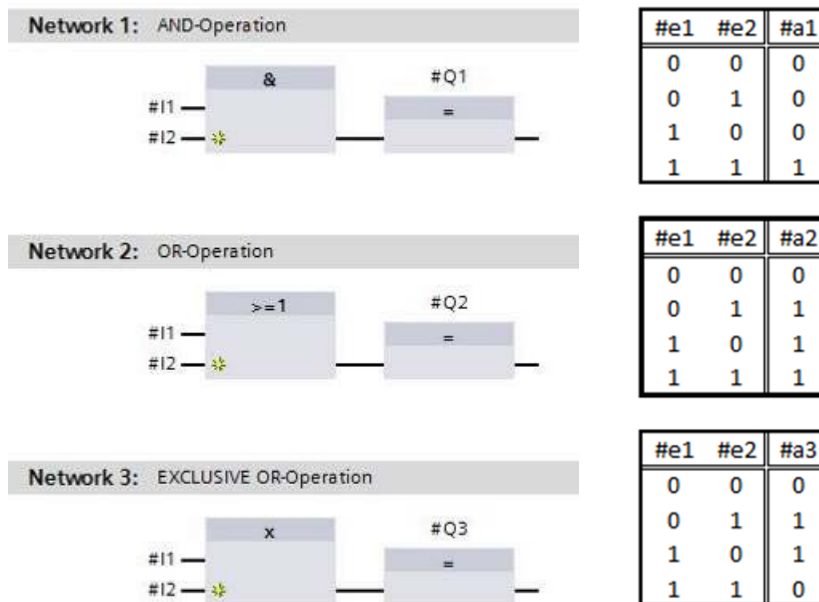


Figure 7: Binary functions in FBD and associated logic table

You can thus use simple instructions, for example, to control binary outputs, evaluate edges and execute jump functions in the program.

Program elements such as IEC timers and IEC counters provide complex instructions.

The empty box serves as a placeholder in which you can select the required instruction.

Enable input EN (enable)/ Enable output ENO (enable output) mechanism:

- An instruction without EN/ENO mechanism is executed independent of the signal state at the box inputs.
- Instructions with EN/ENO mechanism are only executed if enable input "EN input has signal state "1". When the box is processed correctly, enable output "ENO" has signal state "1". As soon as an error occurs during the processing, the "ENO" enable output is reset. If enable input EN is not connected, the box is always executed.

4 Task

The following functions of the sorting station process description will be planned, programmed and tested in this chapter:

- Automatic mode - Conveyor motor

5 Planning

The programming of all functions in OB1 is not recommended for reasons of clarity and reusability. The majority of the program code will therefore be moved into functions (FCs) and function blocks (FBs). The decision on which functions is to be moved to the FB and which is to run in OB 1 is planned below.

5.1 EMERGENCY STOP

The EMERGENCY STOP does not require a separate function. Just like the operating mode, the current state of the EMERGENCY STOP relay can be used directly at the blocks.

5.2 Automatic mode - Conveyor motor

Automatic mode of the conveyor motor is to be encapsulated in a function block (FB) "MOTOR_AUTO". On the one hand, this preserves the clarity of OB1. On the other hand, it enables reuse if another conveyor belt is added to the station. Table 2 lists the planned parameters.

Input	Data	Comment
Automatic_mode_active	BOOL	Automatic mode activated
Start	BOOL	Pushbutton automatic start
Stop	BOOL	Pushbutton automatic stop
Enable_OK	BOOL	All enable conditions OK
Safety_shutoff_active	BOOL	Safety shutoff active, e.g. emergency stop pressed
Output		
Conveyor_motor_automatic_mode	BOOL	Control of the conveyor motor in automatic mode
Static		
Memory_automatic_start_stop	BOOL	Memory used for start/stop automatic mode

Table 2: Parameters for FB "MOTOR_AUTO"

The Memory_automatic_start_stop is latched with Start but only if the reset conditions are not present.

The Memory_automatic_start_stop is reset if Stop is present or safety shutoff is active or automatic mode is not activated (manual mode).

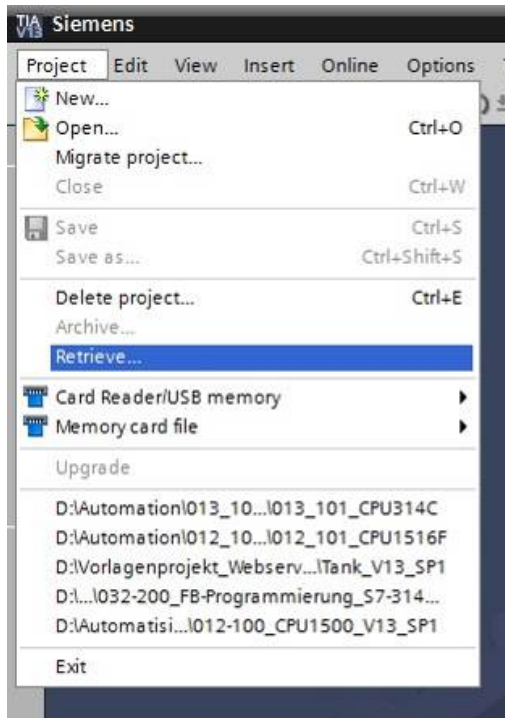
The Conveyor_motor_automatic_mode output is controlled when Memory_automatic_start_stop is set and the enable conditions are met.

6 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

6.1 Retrieve an existing project

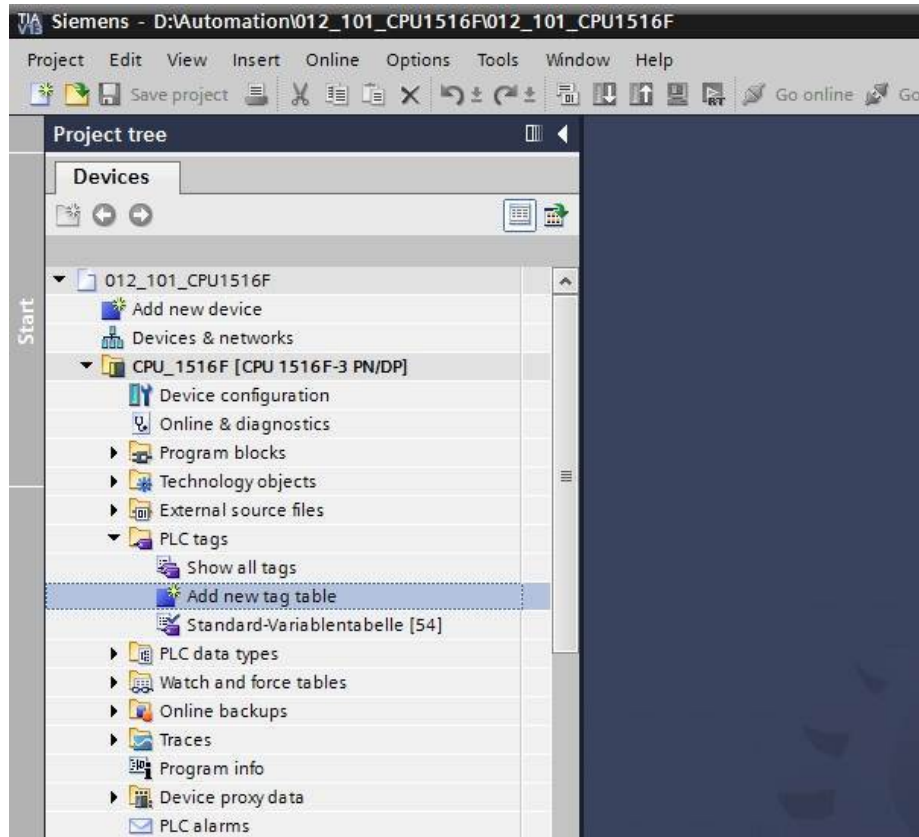
- Before we can start programming the function block (FB) "MOTOR_AUTO", we need a project with a hardware configuration (e.g. SCE_EN_012_101_Hardware_Configuration_S7-1516F_R1502.zap). To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open. (→ Project → Retrieve → Select a .zap archive → Open)



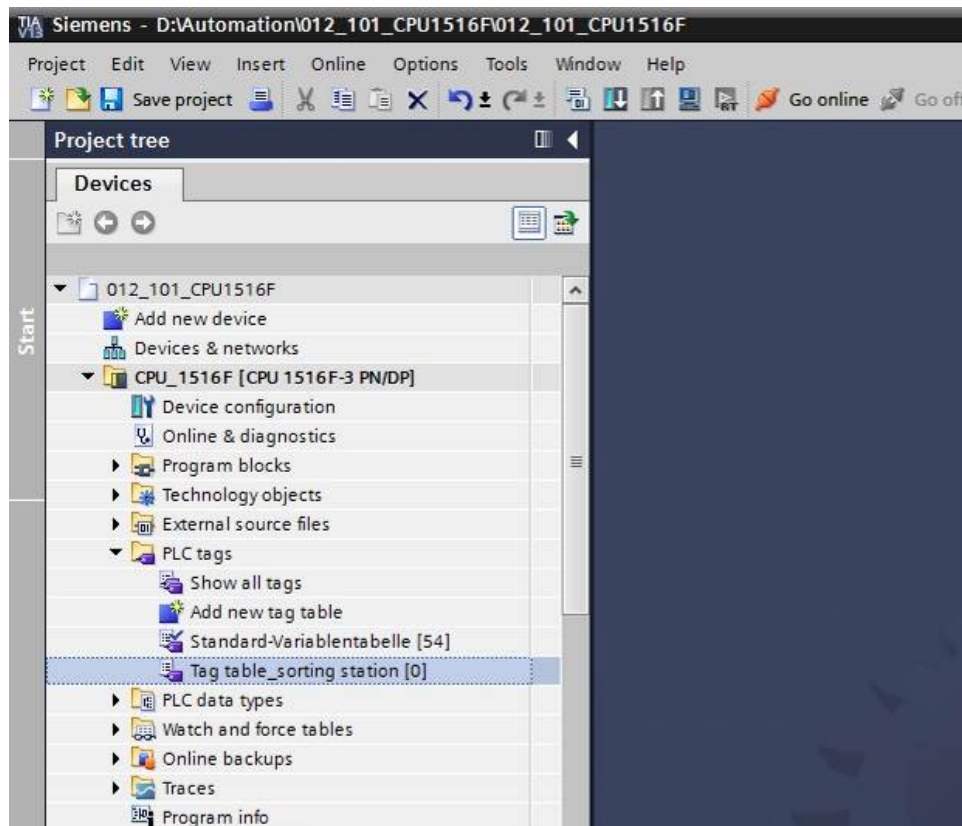
- The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK". (→ Target directory → OK)

6.2 Create a new tag table

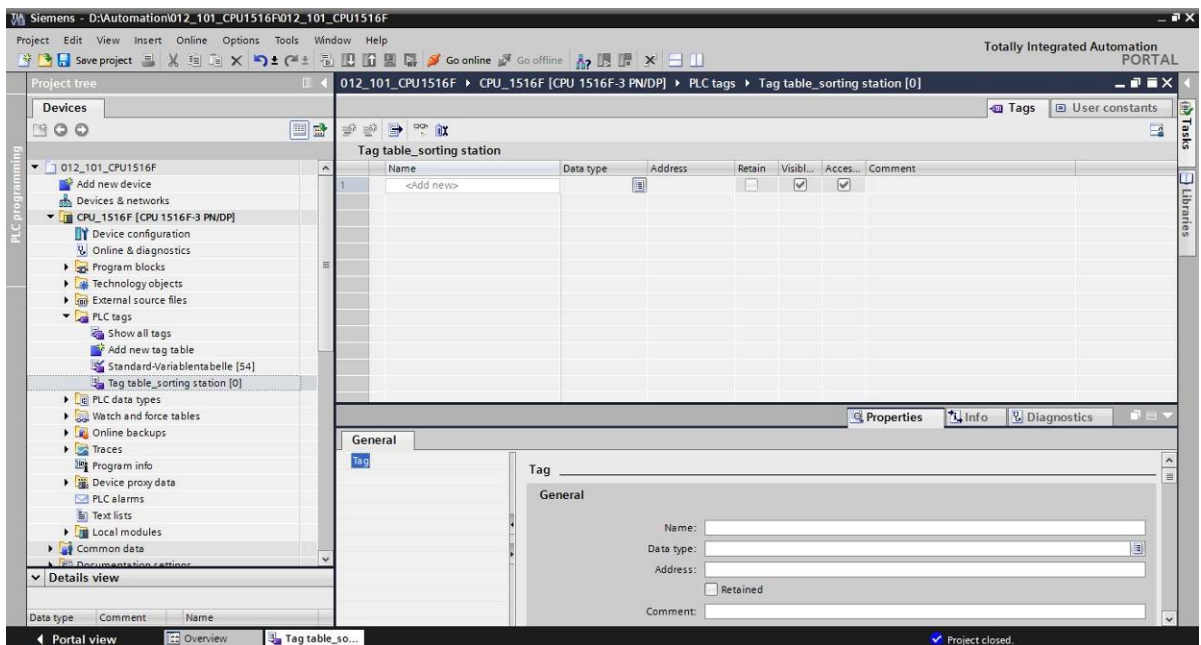
- In the project view, navigate to the → PLC tags of your controller and create a new tag table by double-clicking → Add new tag table.



- Rename the tag table you just created as "Tag_table_sorting_station" (→ right-click "Tag_table_1" → "Rename" → Tag_table_sorting_station).

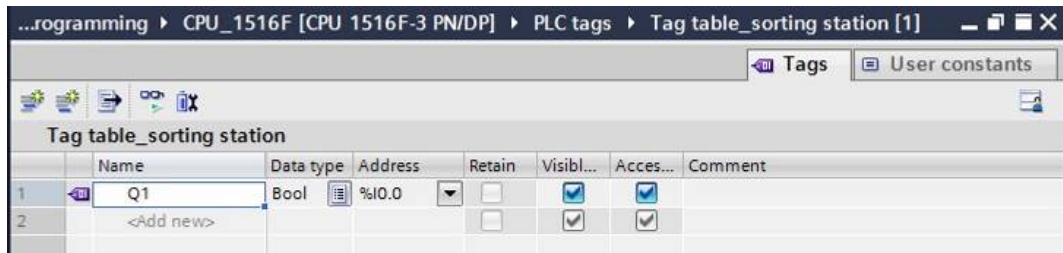


- Open this tag table with a double-click. (→ Tag_table_sorting_station)

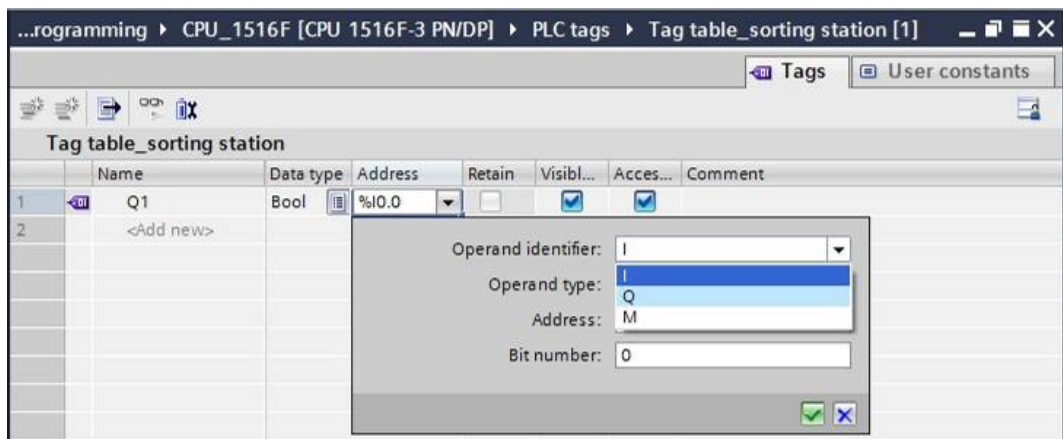


6.3 Create new tags within a tag table

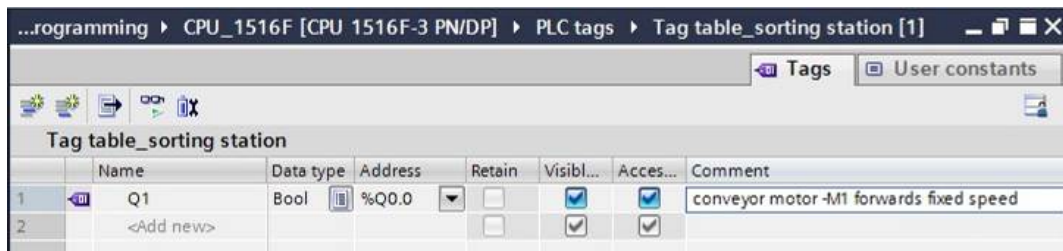
- Add the name Q1 and confirm the entry with the Enter key. If you have not yet created additional tags, TIA Portal now automatically assigns data type "Bool" and address %I0.0 (I 0.0) (→ <Add> → Q1 → Enter).



- Change the address to %Q0.0 (Q 0.0) by entering this directly or by clicking the drop-down arrow to open the Addressing menu, changing the operand identifier to Q and confirming with Enter or by clicking the check mark. (→ %I0.0 → Operand identifier → Q → ☒)



- Enter the "Conveyor motor M1 forwards fixed speed" comment for the tag.



- Add a new Q2 tag in line 2. TIA Portal has automatically assigned the same data type as in line 1 and has incremented the address by 1 to %Q0.1 (Q0.1). Enter the comment "Conveyor motor M1 backwards fixed speed".

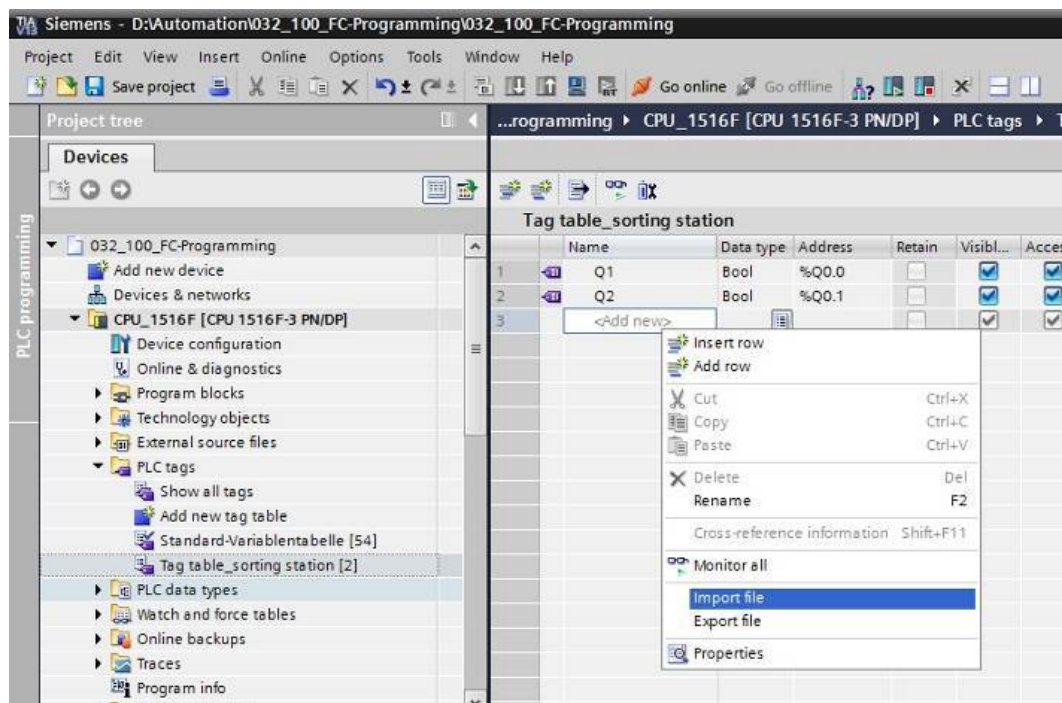
(→ <Add> → Q2 → Enter → Comment → Conveyor motor M1 backwards fixed speed)

	Name	Data type	Address	Retain	Visibl...	Acces...	Comment
1	Q1	Bool	%Q0.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 forwards fixed speed
2	Q2	Bool	%Q0.1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 backwards fixed speed
3	<Add new>			<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

6.4 Import "Tag_table_sorting_station"

- To insert an existing symbol table, right-click on an empty field of the created "Tag_table_sorting_station". Select "Import file" in the shortcut menu.

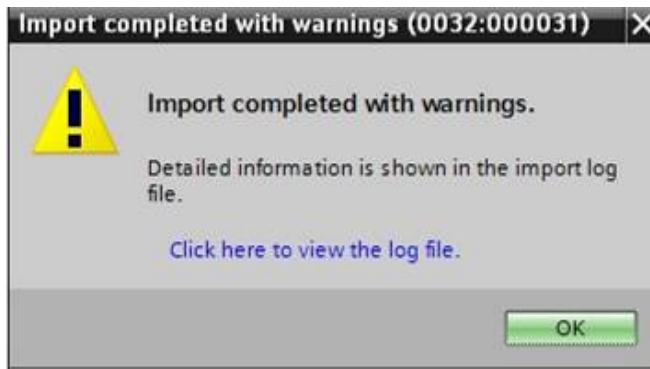
(→ Right-click in an empty field of the tag table → Import file)



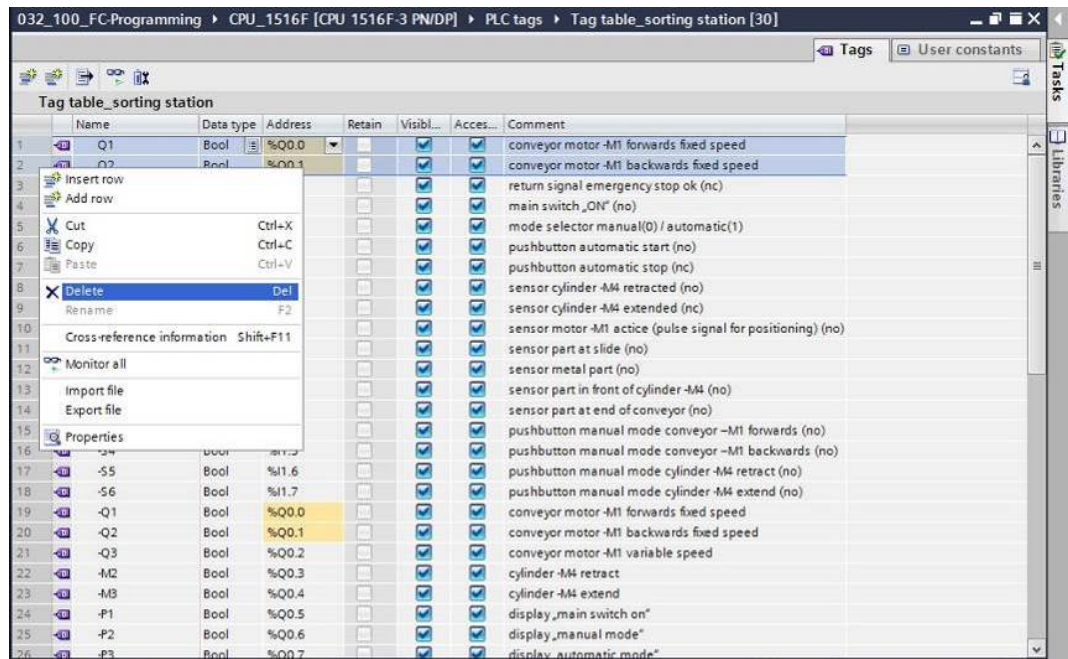
→ Select the desired symbol table (e.g. in .xlsx format) and confirm the selection with "Open".

(→ SCE_EN_020-100_Tag_table_sorting_station... → Open)

→ When the import is finished, you will see a confirmation window and have an opportunity to view the log file for the import. Click → OK.



- You can see that some addresses have been highlighted in orange. These are duplicate addresses and the names of the associated tags have been numbered automatically to avoid confusion.
- Delete the duplicate tags by selecting the lines and pressing the Del key on your keyboard or by selecting "Delete" in the shortcut menu.
- (→ Right-click on selected tags → Delete)



→ You now have a complete symbol table of the digital inputs and outputs in front of you. Save your project under the name 032-100_FCProgramming.


(→ Project → Save as ... → 032-200_FBProgramming → Save)

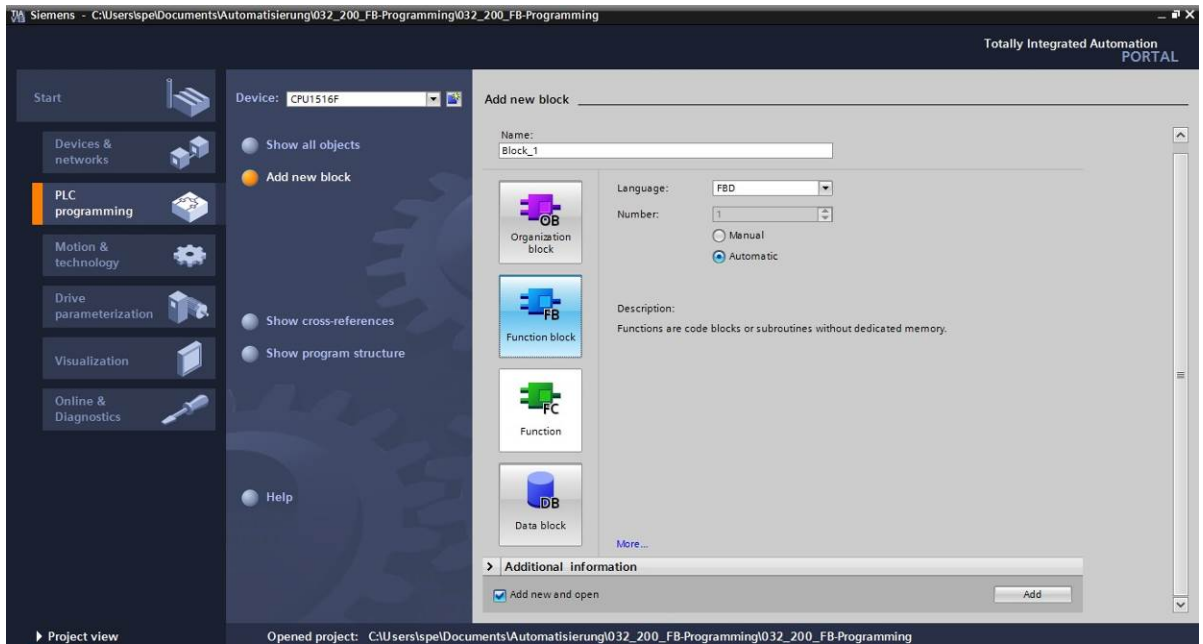
The screenshot shows the Siemens TIA Portal interface. The 'Project' menu is open, displaying options like 'New...', 'Open...', 'Save', 'Save as...', 'Delete project...', 'Archive...', 'Retrieve...', 'Card Reader/USB memory', 'Memory card file', 'Upgrade', 'Print...', and 'Print preview...'. The main window displays the 'Tag table_sorting station' table, which contains 26 rows of data. The table columns are: Name, Data type, Address, Retain, Visibl., Acces., and Comment. The data is as follows:

Name	Data type	Address	Retain	Visibl.	Acces.	Comment
-A1	Bool	%I0.0				return signal emergency stop ok (nc)
-X0	Bool	%I0.1				main switch „ON“ (no)
-S0	Bool	%I0.2				mode selector manual(0) / automatic(1)
-S1	Bool	%I0.3				pushbutton automatic start (no)
-S2	Bool	%I0.4				pushbutton automatic stop (nc)
-B1	Bool	%I0.5				sensor cylinder -M4 retracted (no)
-B2	Bool	%I0.6				sensor cylinder -M4 extended (nc)
-B3	Bool	%I0.7				sensor motor -M1 actice (pulse signal for ...)
-B4	Bool	%I1.0				sensor part at slide (no)
-B5	Bool	%I1.1				sensor metal part (no)
-B6	Bool	%I1.2				sensor part in front of cylinder -M4 (no)
-B7	Bool	%I1.3				sensor part at end of conveyor (no)
-S3	Bool	%I1.4				pushbutton manual mode conveyor -M1...
-S4	Bool	%I1.5				pushbutton manual mode conveyor -M1...
-S5	Bool	%I1.6				pushbutton manual mode cylinder -M4 re...
-S6	Bool	%I1.7				pushbutton manual mode cylinder -M4 ex...
-Q1	Bool	%Q0.0				conveyor motor -M1 forwards fixed speed
-Q2	Bool	%Q0.1				conveyor motor -M1 backwards fixed speed
-Q3	Bool	%Q0.2				conveyor motor -M1 variable speed
-M2	Bool	%Q0.3				cylinder -M4 retract
-M3	Bool	%Q0.4				cylinder -M4 extend
-P1	Bool	%Q0.5				display „main switch on“
-P2	Bool	%Q0.6				display „manual mode“
-P3	Bool	%Q0.7				display „automatic mode“
-P4	Bool	%Q1.0				display „emergency stop activated“
-P5	Bool	%Q1.1				display „automatic mode started“

6.5 Create function block FB1 "MOTOR_AUTO" for the conveyor motor in automatic mode

- In the PLC programming section of the portal view, click "Add new block" to create a new function block.

(→ PLC programming → Add new block → )



→ Rename your new block to: "MOTOR_AUTO", set the language to FBD and keep automatic assignment of the number. Select the "Add new and open" check box. You are then taken automatically to your created function block in the project view. Click "Add".

(→ Name: MOTOR_AUTO → Language: FBD → Number: Automatic → ☒ Add new and open → Add)

Add new block

Name:

Organization block (OB)

Function block (FB)

Function (FC)

Data block (DB)

Language:

Number:

☐ Manual

☒ Automatic

Description:
Function blocks are code blocks that store their values permanently in instance data blocks, so that they remain available after the block has been executed.

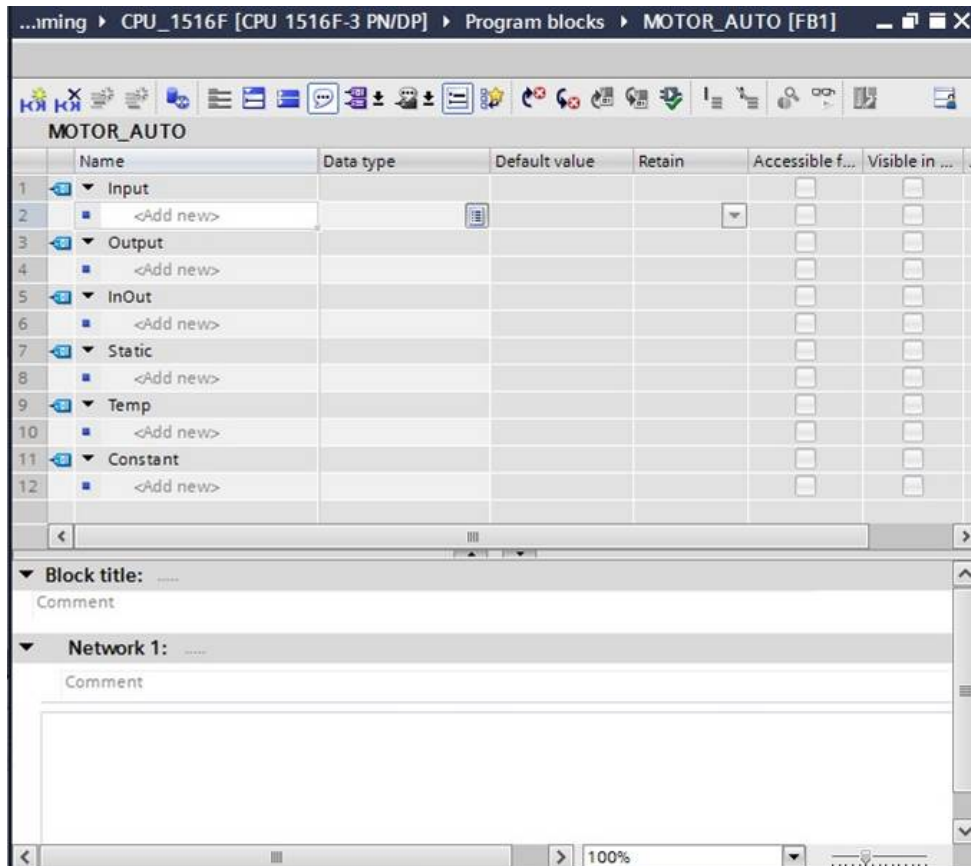
[More...](#)

> **Additional information**

☒ Add new and open

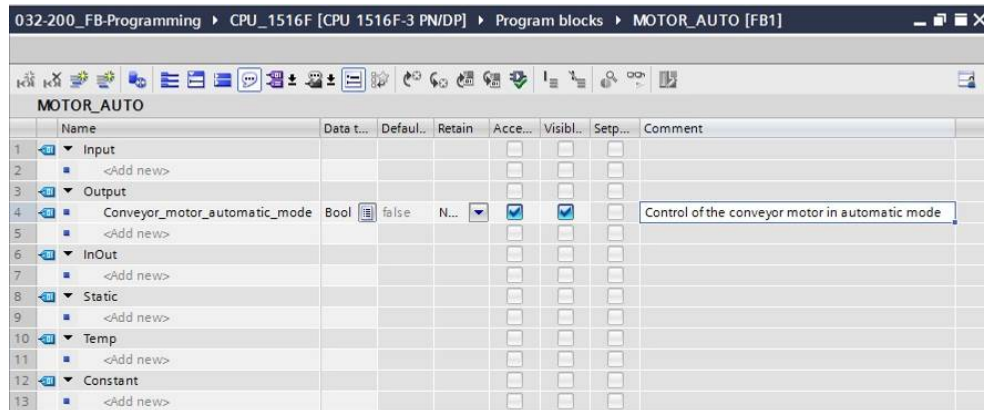
6.6 Define the interface of FB1 "MOTOR_AUTO"

- If you selected "Add new and open", the project view opens with a window for creating the block you just added.
- You can find the interface description of your function block in the upper section of your programming view.



- A binary output signal is needed for controlling the conveyor motor. For this reason, we first create local output tag #Conveyor_motor_automatic_mode of the "Bool" type. Enter the comment "Control of the conveyor motor in automatic mode" for the parameter.

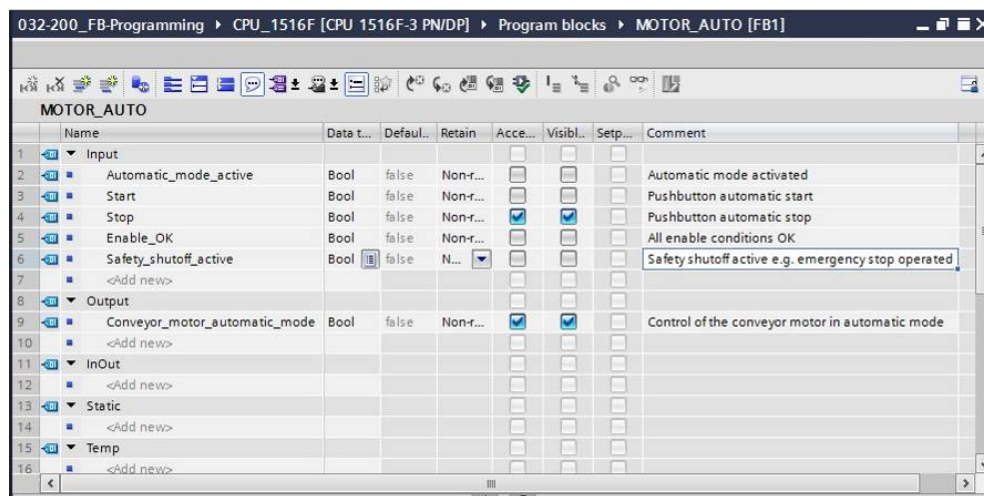
(→ Output: Conveyor_motor_automatic_mode → Bool → Control of the conveyor motor in automatic mode)



- Add parameter #Automatic_mode_active as the input interface under Input and confirm the entry with the Enter key or by exiting the entry field. Data type "Bool" is assigned automatically. This will be retained. Next, enter the associated comment "Automatic mode activated".

(→ Automatic_mode_active → Bool → Automatic mode activated)

- Add parameters #Start, #Stop, #Enable_OK and #Safety_shutoff_active as additional binary input parameters under Input and check their data types. Add descriptive comments.



The conveyor is started and stopped with pushbuttons. We therefore need a "Static" tag as a memory. Under Static, add tag #Memory_automatic_start_stop and confirm the entry with the Enter key or by exiting the entry field. Data type "Bool" is assigned automatically. This will be retained. Enter the associated comment "Memory used for start/stop automatic mode". (→ Memory_automatic_start_stop → Bool → Memory used for start/stop automatic mode)

032-200_FB-Programming ▶ CPU_1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ MOTOR_AUTO [FB1]

MOTOR_AUTO

	Name	Data t...	Default..	Retain	Acce...	Visibl..	Setp...	Comment
1	▼ Input							
2	Automatic_mode_active	Bool	false	Non-r...				Automatic mode activated
3	Start	Bool	false	Non-r...				Pushbutton automatic start
4	Stop	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Pushbutton automatic stop
5	Enable_OK	Bool	false	Non-r...				All enable conditions OK
6	Safety_shutoff_active	Bool	false	Non-r...				Safety shutoff active e.g. emergency stop operated
7	<Add new>							
8	▼ Output							
9	Conveyor_motor_automatic_mode	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Control of the conveyor motor in automatic mode
10	<Add new>							
11	▼ InOut							
12	<Add new>							
13	▼ Static							
14	Memory_automatic_start_stop	Bool	false	N...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Memory used for start/ stop automatic mode

→ For purposes of program documentation, assign the block title, a block comment and a helpful network title for Network 1.

(→ Block title: Motor control in automatic mode → Network 1: Memory_automatic_start_stop and control of the conveyor motor in automatic mode)

MOTOR_AUTO

	Name	Data t...	Default..	Retain	Acce...	Visibl..	Setp...	Comment
1	▼ Input							
2	Automatic_mode_active	Bool	false	Non-r...				Automatic mode activated
3	Start	Bool	false	Non-r...				Pushbutton automatic start
4	Stop	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Pushbutton automatic stop
5	Enable_OK	Bool	false	Non-r...				All enable conditions OK
6	Safety_shutoff_active	Bool	false	Non-r...				Safety shutoff active e.g. emergenc...
7	▼ Output							
8	Conveyor_motor_automatic_mode	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Control of the conveyor motor in a...
9	▼ InOut							
10	<Add new>							
11	▼ Static							
12	Memory_automatic_start_stop	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Memory used for start/ stop autom...

▼ **Block title:** Motor control in automatic mode

▼ **Conveyor motor in automatic mode:**
 The bit Memory_automatic_start_stop is set with the input Start, but only if the reset conditions are not fulfilled.
 The bit Memory_automatic_start_stop is reset with the input Stop or if the safety shutoff is activated or if the automatic mode is not activated (manual mode).
 If Memory_automatic_start_stop is set, the enable conditions are granted and Memory_conveyor_start_stop is set the output Conveyor_motor_automatic_mode is activated. For reasons of energy efficiency the conveyor motor should only run if a part is present. Therefore Memory_conveyor_start_stop is set if there is a part detected in front of Sensor_slide and reset with a negative edge at Sensor_end_of_conveyor or if the safety shutoff is activated or if the automatic mode is not activated (manual mode).

▼ **Network 1:** Memory automatic_start_stop and control of the conveyor motor in automatic mode

100%

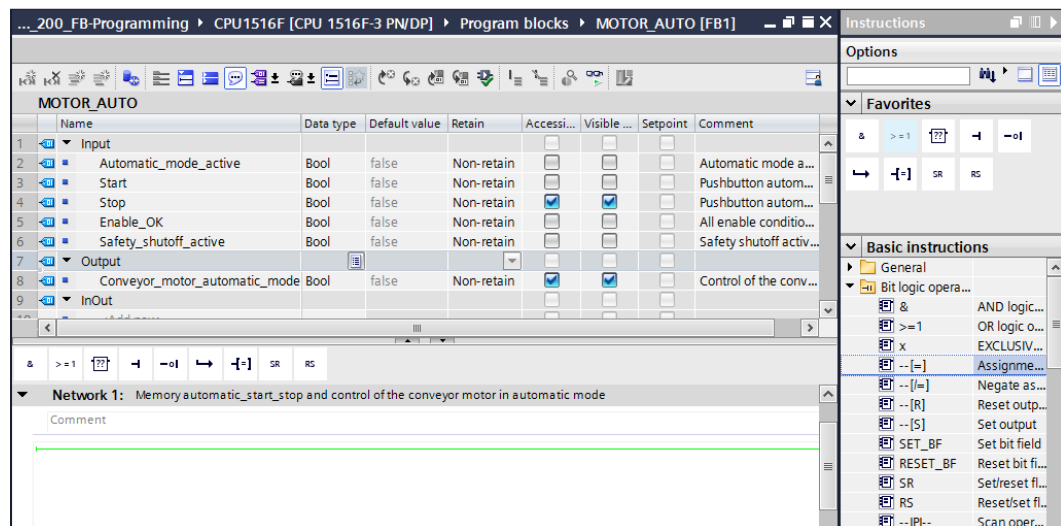
6.7 Program FB1: MOTOR_AUTO


- Below the interface description, you see a toolbar in the programming window with various logic functions and below that an area with networks. We have already specified the block title and the title for the first network there. Programming is performed within the networks using individual logic blocks. Distribution among multiple networks helps to preserve the clarity of the program. In the following, you will get to know the various ways you can insert logic blocks.




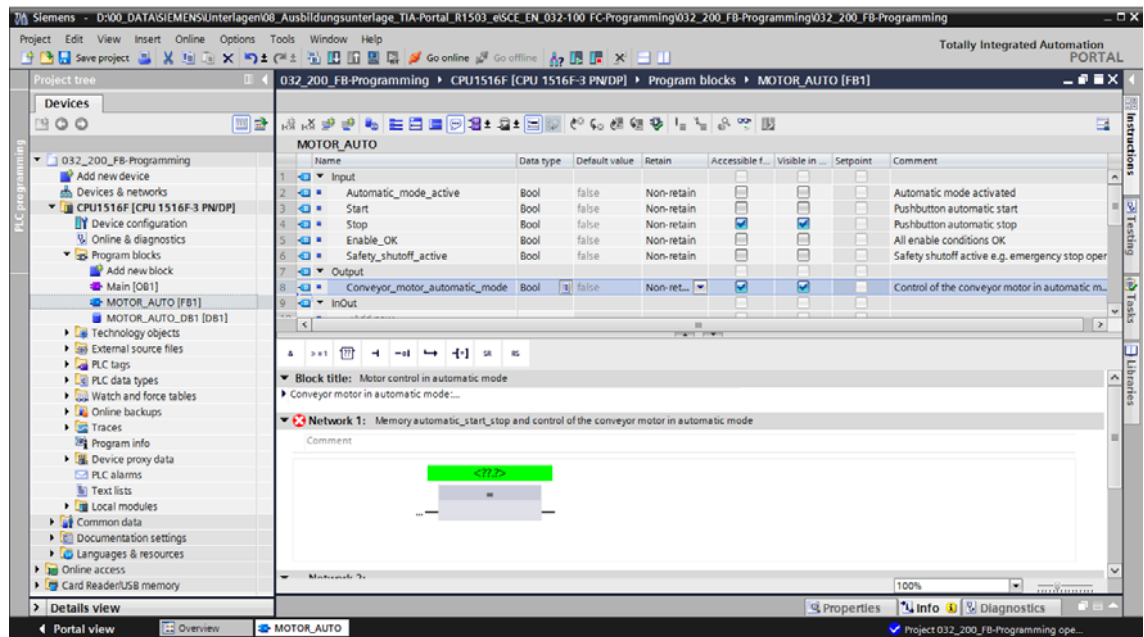
- You can see a list of instructions you can use in the program on the right side of your programming window. Under → Basic instructions → Bit logic operations, find function (Assignment) and use a drag-and-drop operation to move it to Network 1 (green line appears, mouse pointer with + symbol).

(→ Instructions → Basic instructions → Bit logic operations →)




→ Now use drag-and-drop to move your output parameter #Conveyor_motor_automatic_mode onto $\langle ?? \rangle$ above the block you just inserted. The best way to select a parameter in the interface description is by "grabbing" it at the blue symbol .

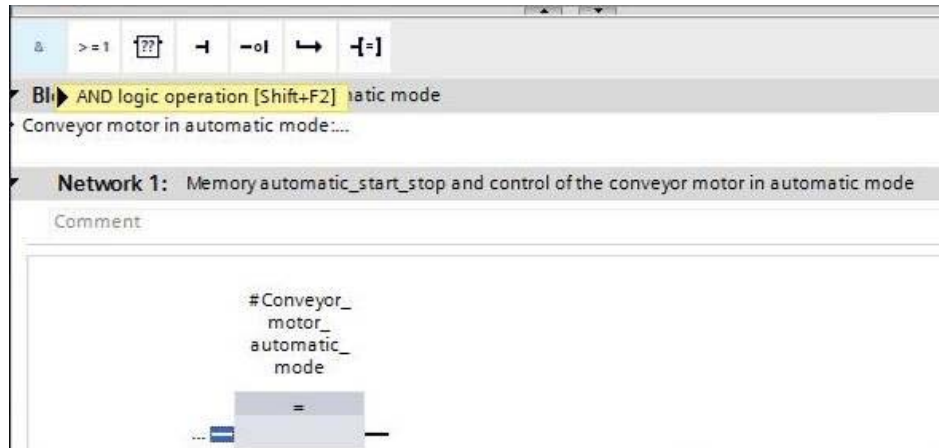
(→  Conveyor_motor_automatic_mode)




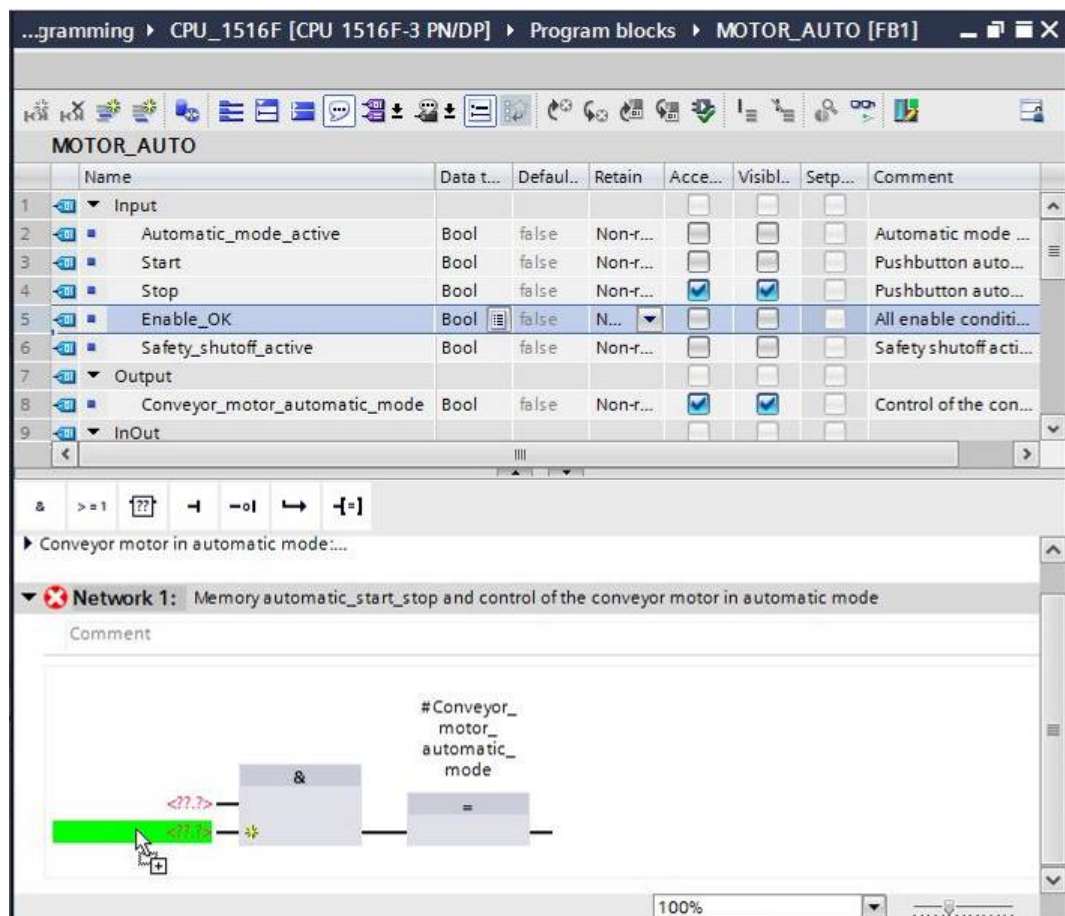
→ This determines that the #Conveyor_motor_automatic_mode parameter is written by this block. Still missing, however, are the input conditions so that this actually happens. An SR flip-flop and #Enable_OK parameter are logically combined with an AND logic operation at the input of the assignment block. To do this, first click the input of the block so that the input line has a blue background.







- Click the  icon in your logic toolbar to insert an AND logic operation before your assignment block.

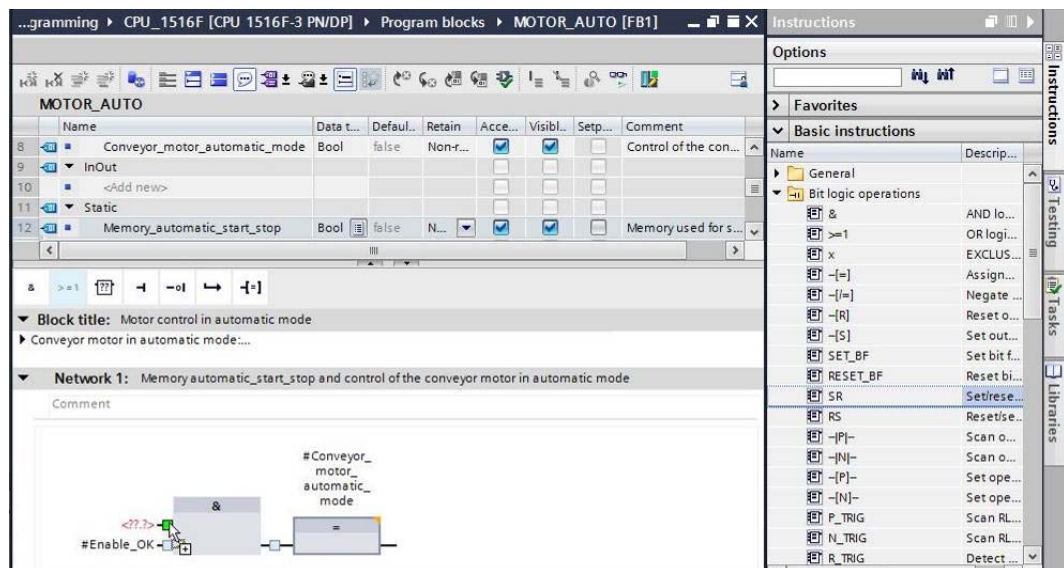




- Use drag-and-drop to move input parameter #Enable_OK onto the second input of the & logic operation <??.?>. (→  Enable_OK)

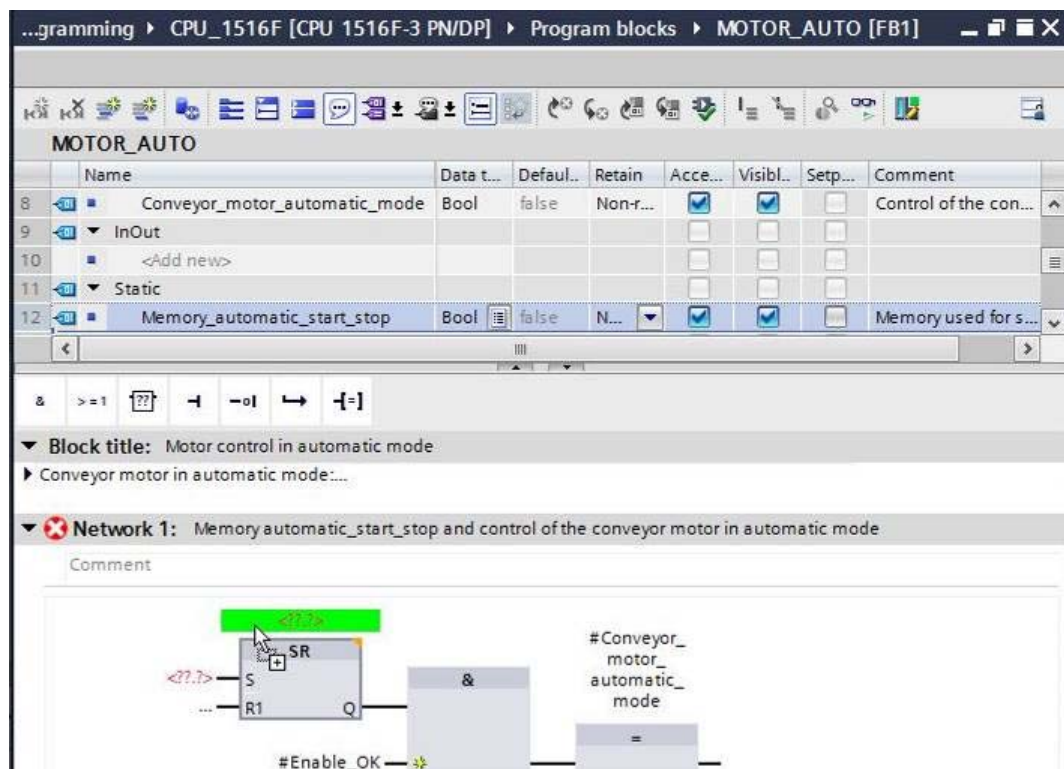


→ Use drag-and-drop to move the Set/reset flip-flop function  from the list of instructions under → Basic instructions → Bit logic operations onto the first input of the & operation .

(→ Instructions → Basic instructions → Bit logic operations →  → )

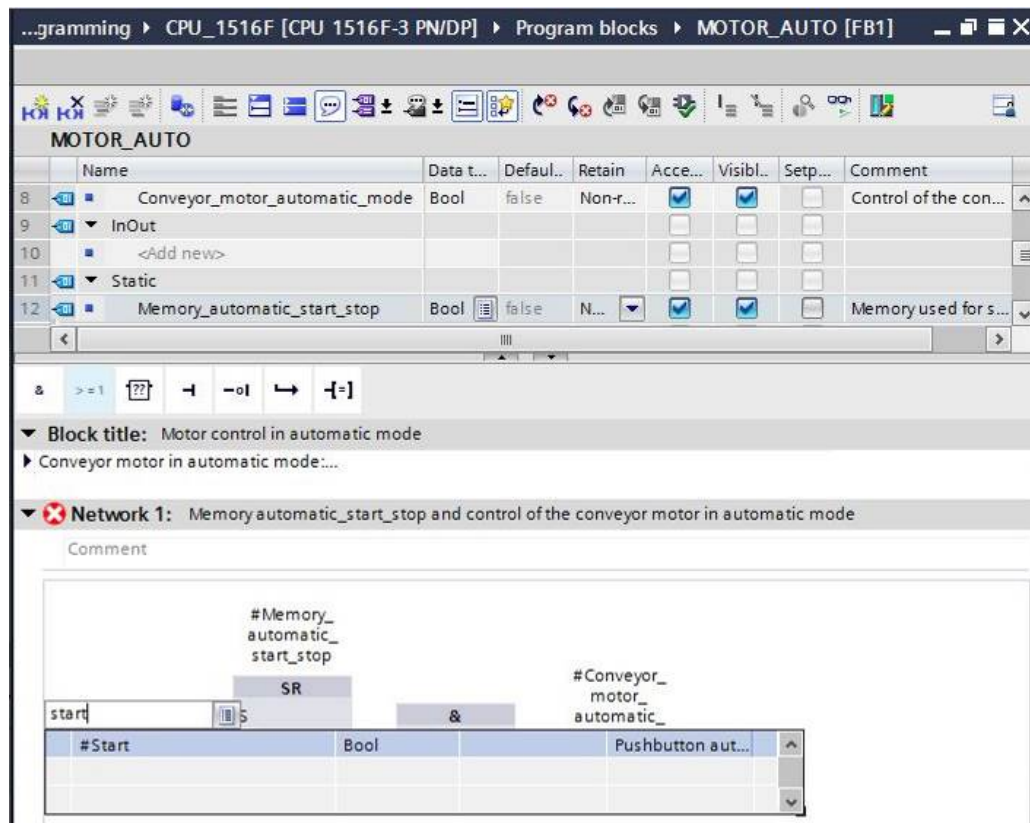


→ The SR flip-flop requires a memory tag. For this, use drag-and-drop to move static parameter #Memory_automatic_start_stop onto the  above the SR flip-flop. (→ )



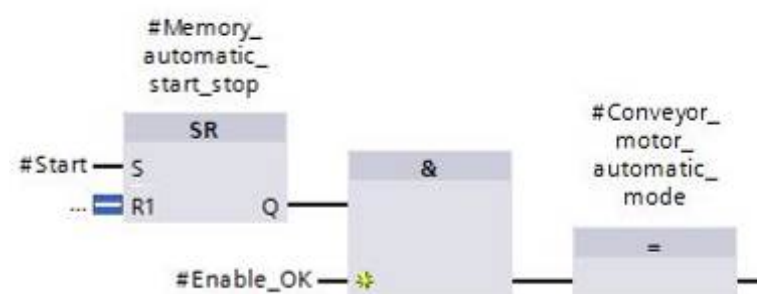
- The #Memory_automatic_start_stop will be set with input tag #Start. Click twice on the S input of the SR flip-flop <???.?> and enter "Start" in the field that appears in order to see a list of available tags starting with "Start". Click the #Start tag and apply with → Enter.

(→ SR flip-flop → <???.?> → Start → #Start → Enter)

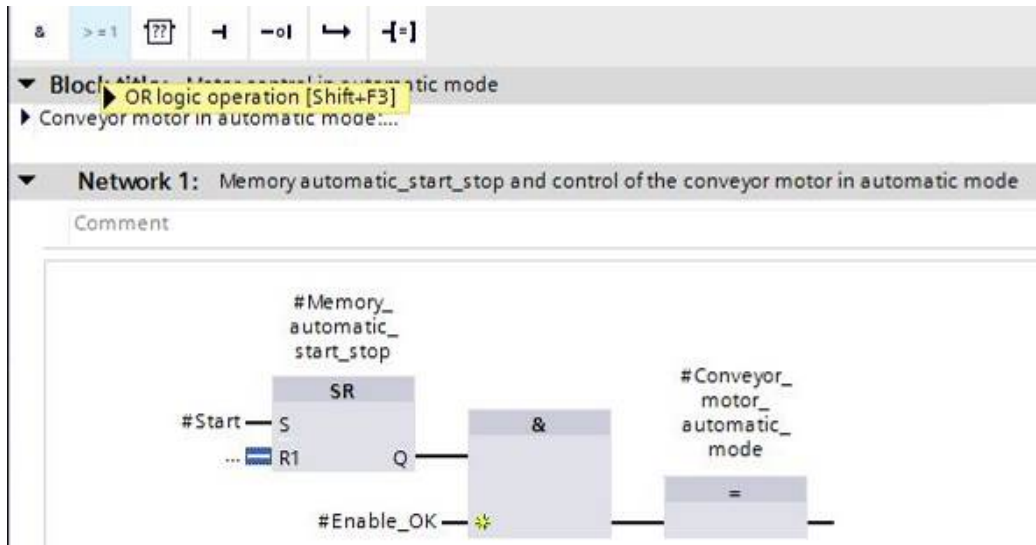



Note: When assigning tags in this way, there is a risk of a mix-up with the global tags from the tag table. The previously presented procedure using drag and drop from the interface description should therefore be used preferentially.

- Multiple conditions are to be able to stop the conveyor. An OR block is therefore needed at the R1 input of the SR flip-flop. First, click the R1 input of the SR flip-flop so that the input line has a blue background.



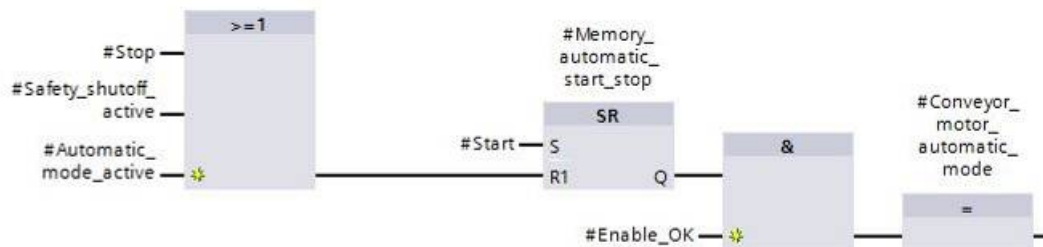
→ Click the ≥ 1 icon in your logic toolbar to insert an OR logic operation.




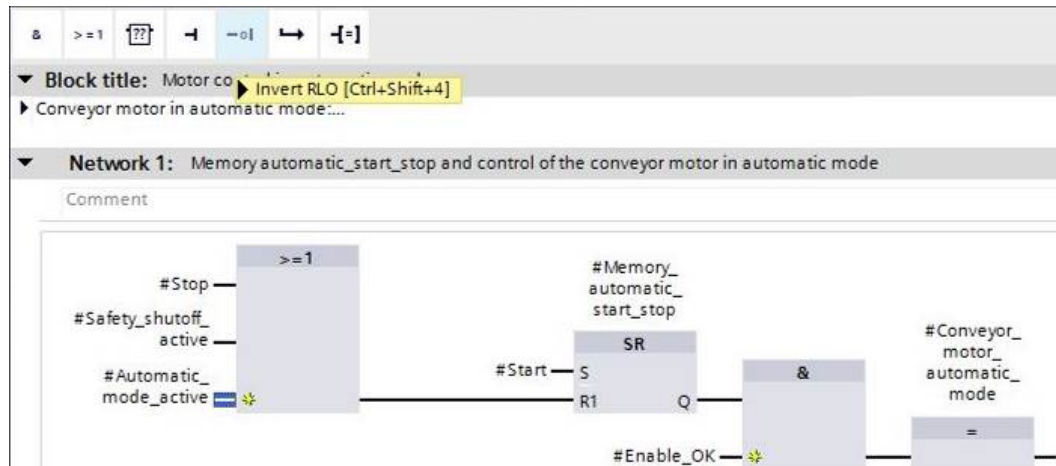
→ The OR block has 2 inputs initially. In order to logically combine an additional input tag, click the yellow star  of the OR block.




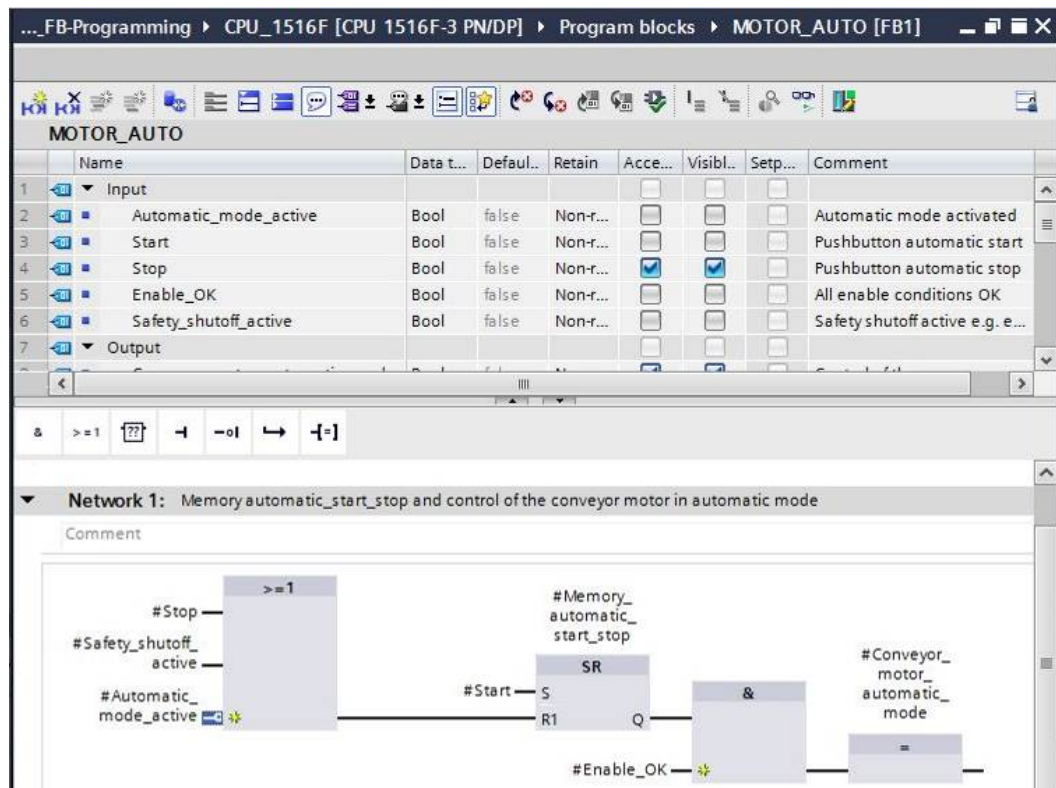
→ Add input tags #Stop, #Safety_shutoff_active and #Automatic_mode_active to the 3 inputs of the OR block.



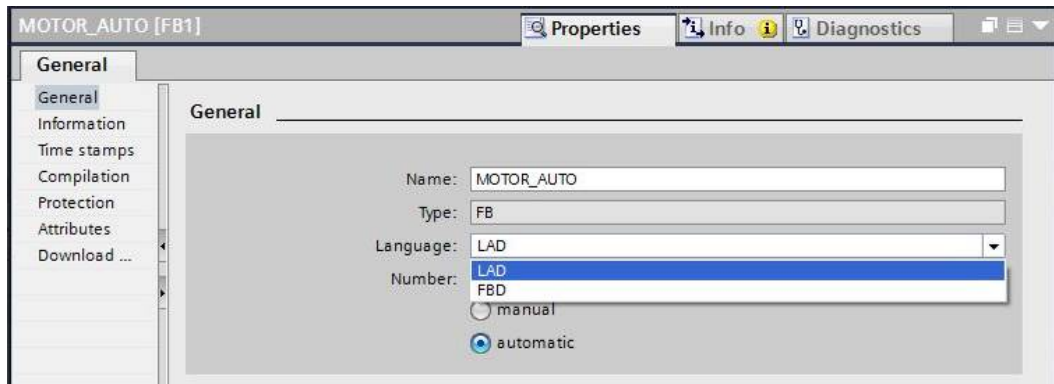
- Negate the input connected to parameter #Automatic_mode_active by selecting it and clicking .



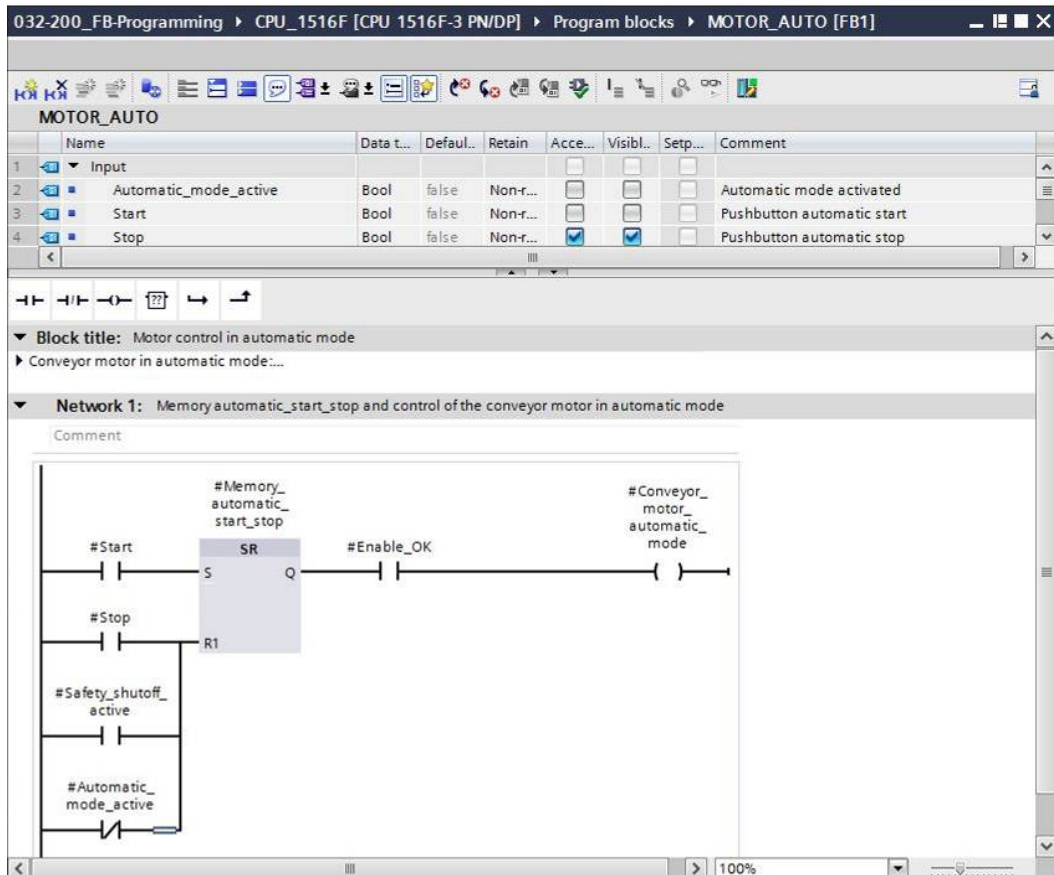
- Do not forget to click  **Save project**. The finished function block "MOTOR_AUTO" [FB1] in FBD is shown below.



→ Under "General" in the properties of the block, you can change the "Language" to LAD (Ladder Logic) (→ Properties → General → Language: LAD)



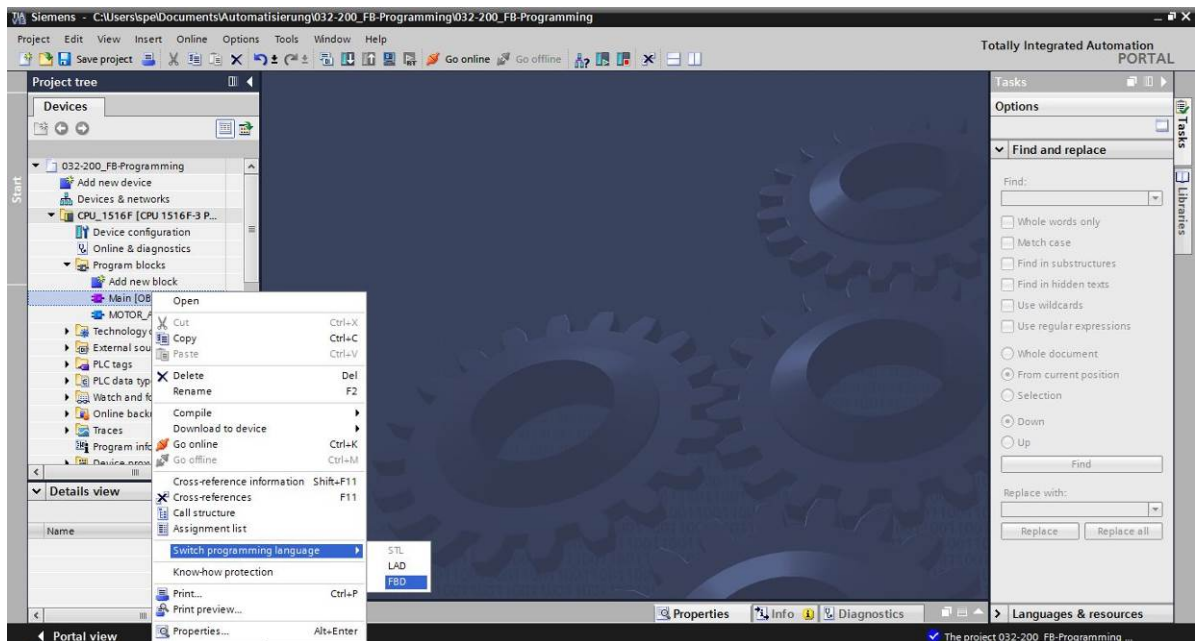
→ The program has the following appearance in LAD.



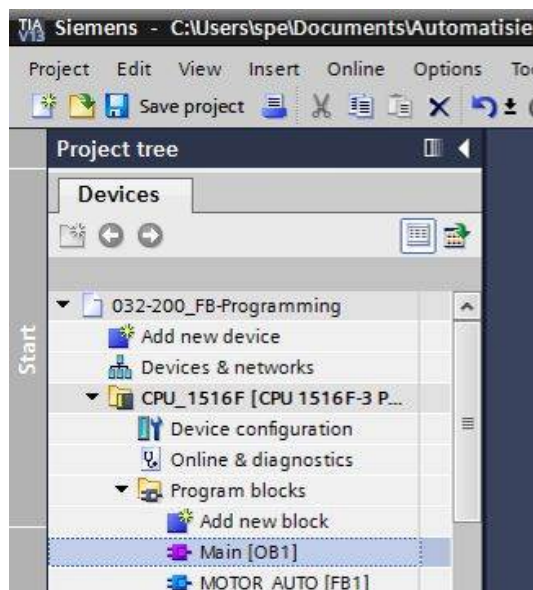
6.8 Program organization block OB1 – Control of the forward belt tracking in automatic mode

→ Before programming organization block "Main [OB1]", we switch the programming language to FBD (Function Block Diagram). To do so, first click on "Main [OB1]" in the "Program blocks" folder.

(→ CPU_1516F[CPU 1516F-3 PN/DP → Program blocks → Main [OB1] → Switch programming language → FBD)

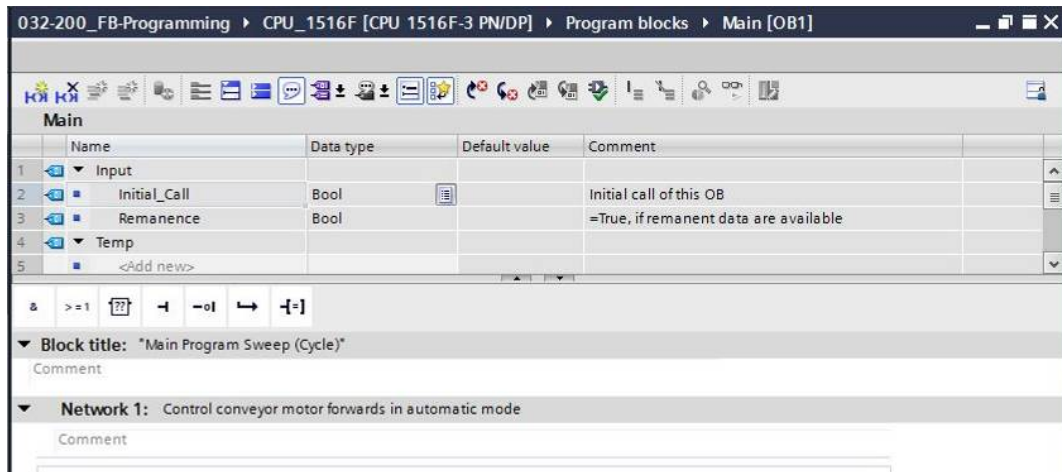


→ Open the "Main [OB1]" organization block with a double-click.

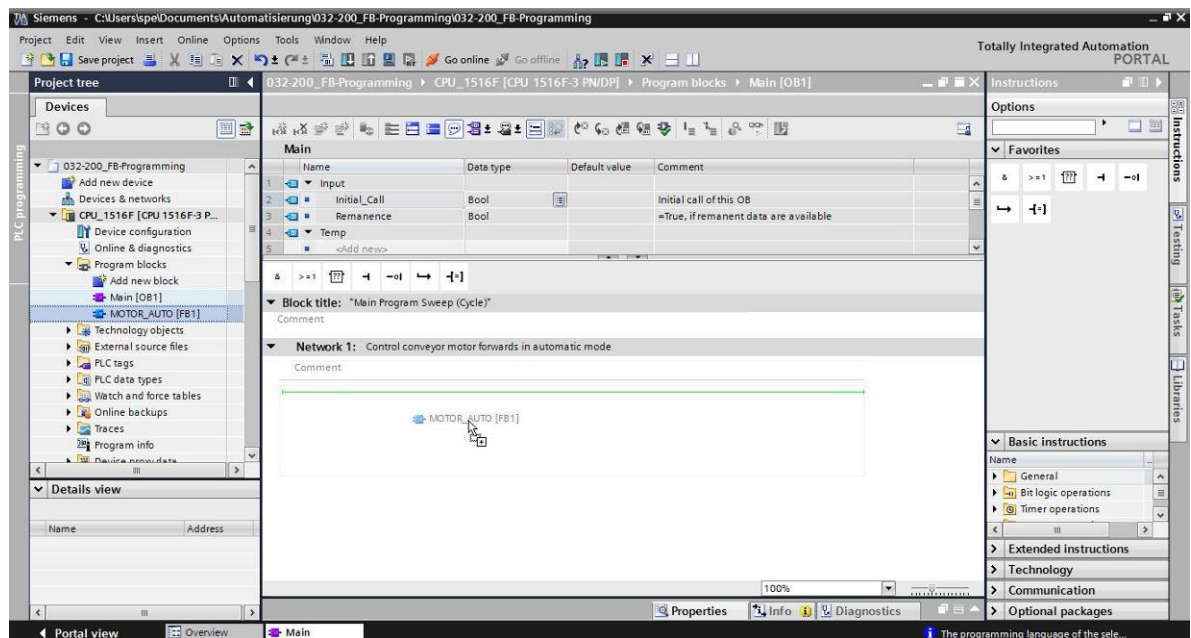


→ Assign Network 1 the name "Control conveyor tracking forward in automatic mode"

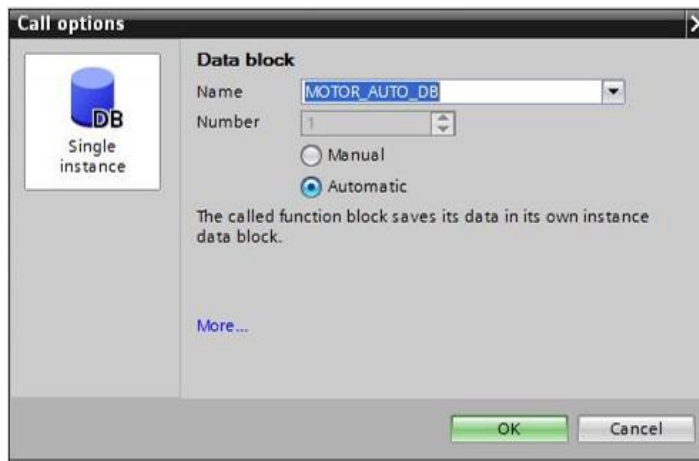
(→ Network 1:... →Control conveyor motor forwards in automatic mode)



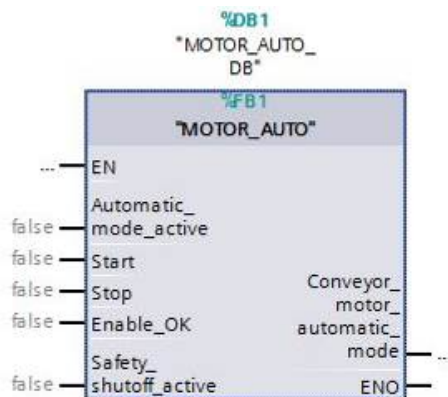
→ Use drag-and-drop to move your "MOTOR_AUTO [FB1]" function block onto the green line in Network 1.





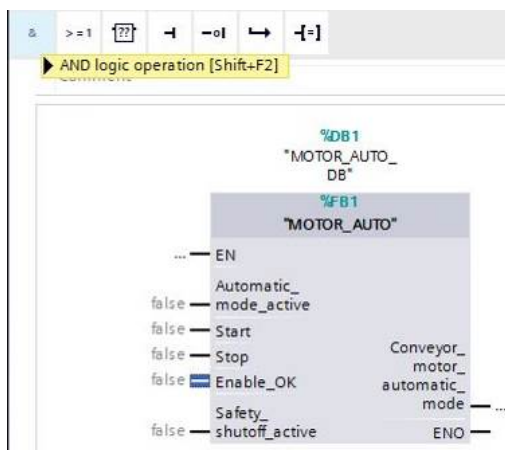
- The instance data block for this call of FB1 is created automatically. Assign a name and apply it with OK. (→ MOTOR_AUTO_DB1 → OK)



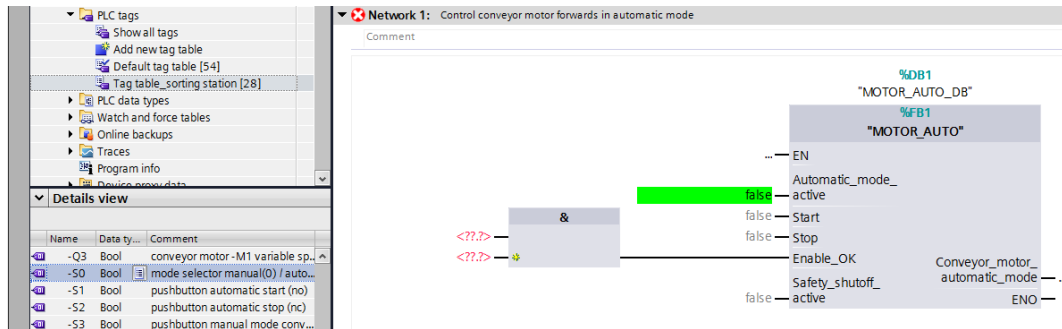
- A block with the interface you defined, the instance data block and connections EN and ENO are inserted in Network 1.



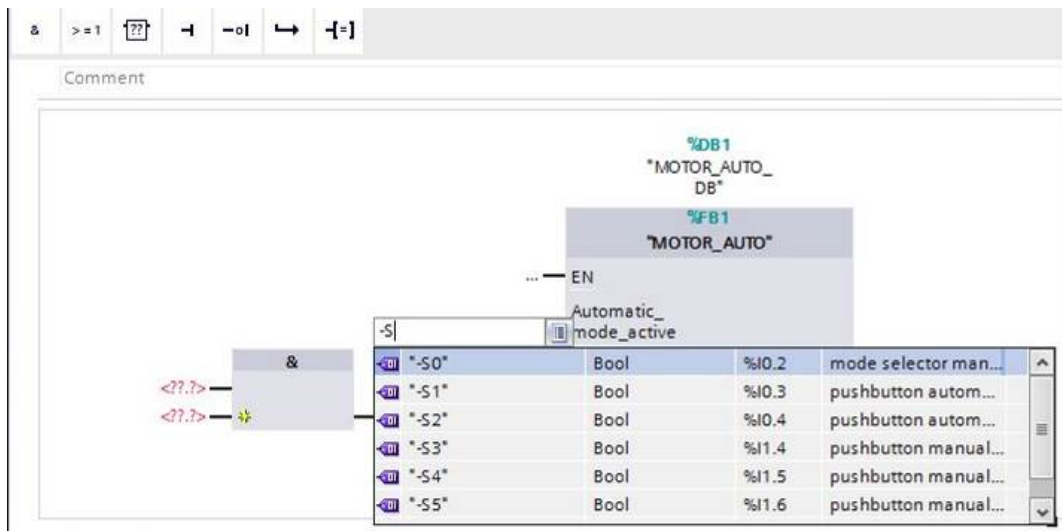
- To insert an AND before input parameter "Enable_OK", select this input and insert the AND by clicking the  icon in your logic toolbar (→ ).



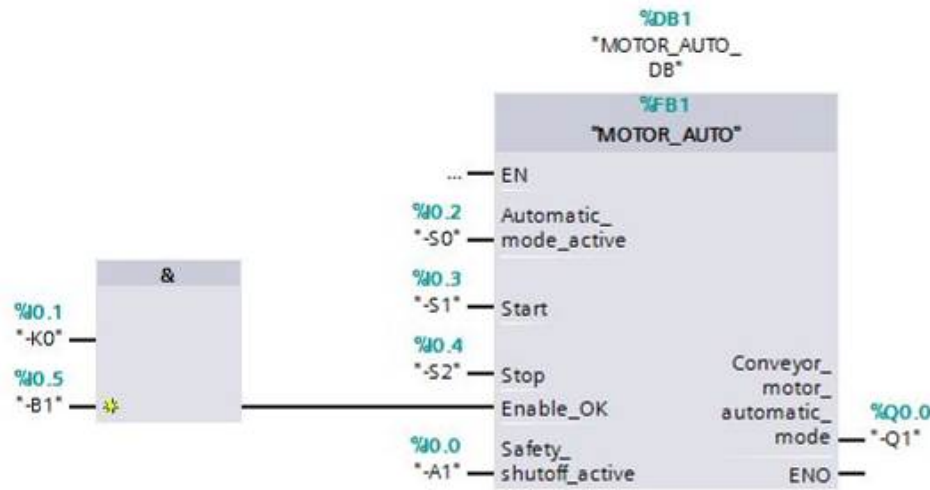
- To connect the block to the global tags from "Tag_table_sorting_station", we have two options:
- Either select the "Tag_table_sorting_station" in the project tree and use drag-and-drop to move the desired global tag from the Details view to the interface of FC1 (→ Tag_table_sorting_station → Details view. → S0 → Automatic_mode_active)




- Or, enter the starting letters (e.g. "S") "-S" of the desired global tag for '<???.?>' and select the global input tag "-S0" (%I0.2) from the displayed list. (→ Automatic_mode_active → S → -S0)

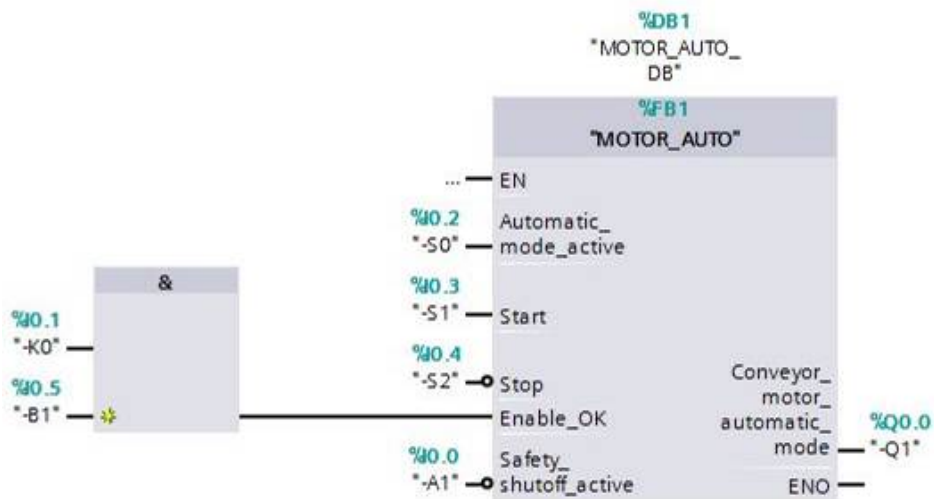


- Insert the other input tags "-S1", "-S2", "-K0", "-B1" and "-A1" and insert output tag "-Q1" (%Q0.0) at output "Conveyor_motor_automatic_mode".

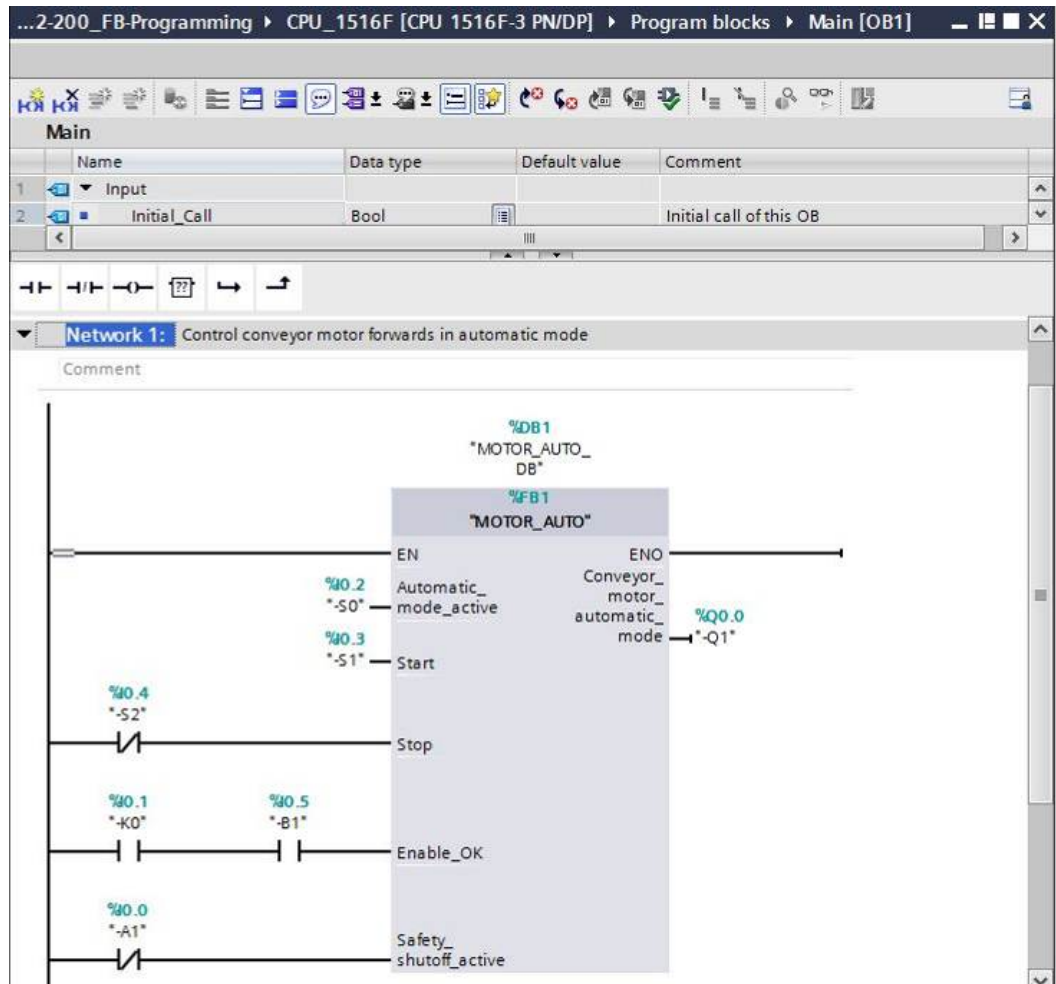


- Negate the querying of input tags "-S2" and "-A1" by selecting them and clicking .





(→ -S2 →  → -A1 → )

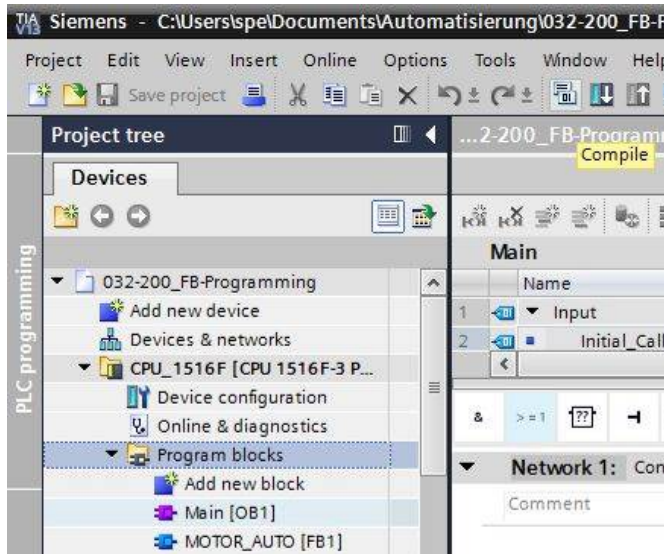


6.9 The result in the LAD (Ladder Logic) programming language has the following appearance.

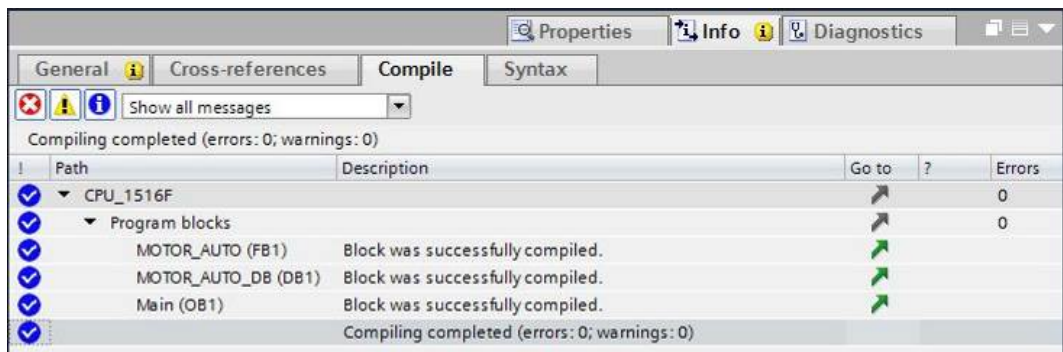


6.10 Save and compile the program

- To save your project, select the  **Save project** button in the menu. To compile all blocks, click the "Program blocks" folder and select the  icon for compiling in the menu (→  **Save project** → Program blocks → ).



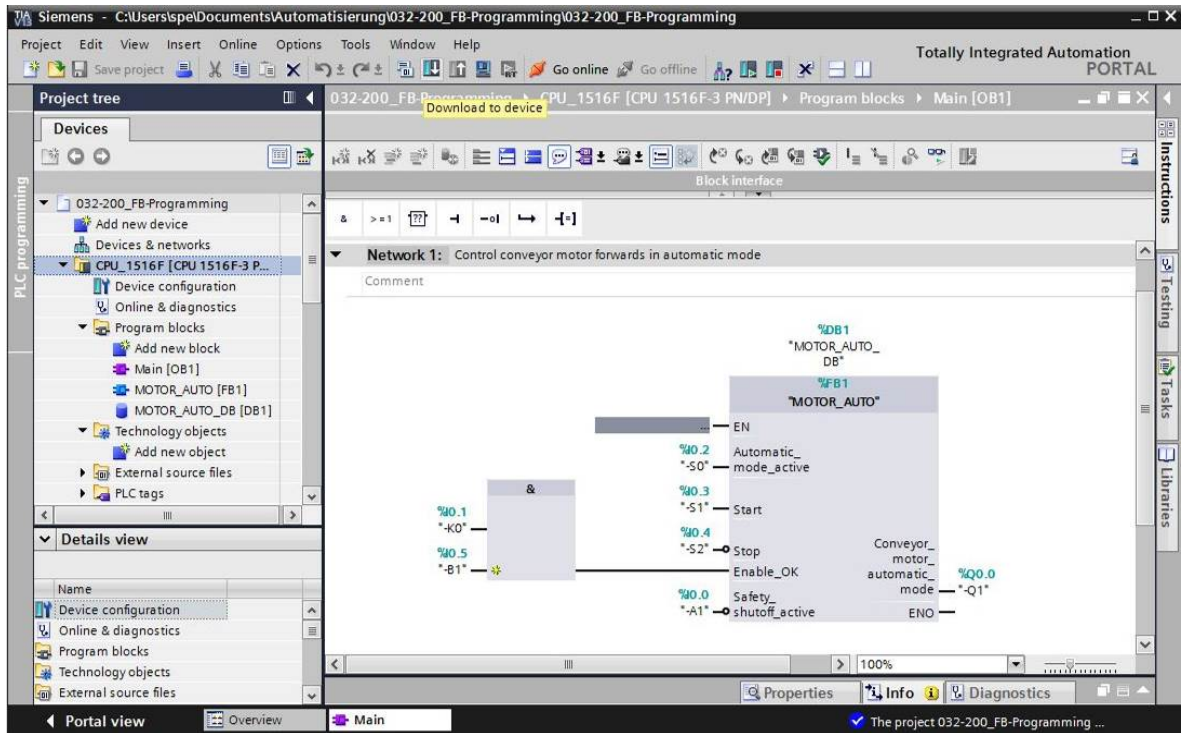
- The "Info", "Compile" area shows which blocks were successfully compiled.





6.11 Download the program

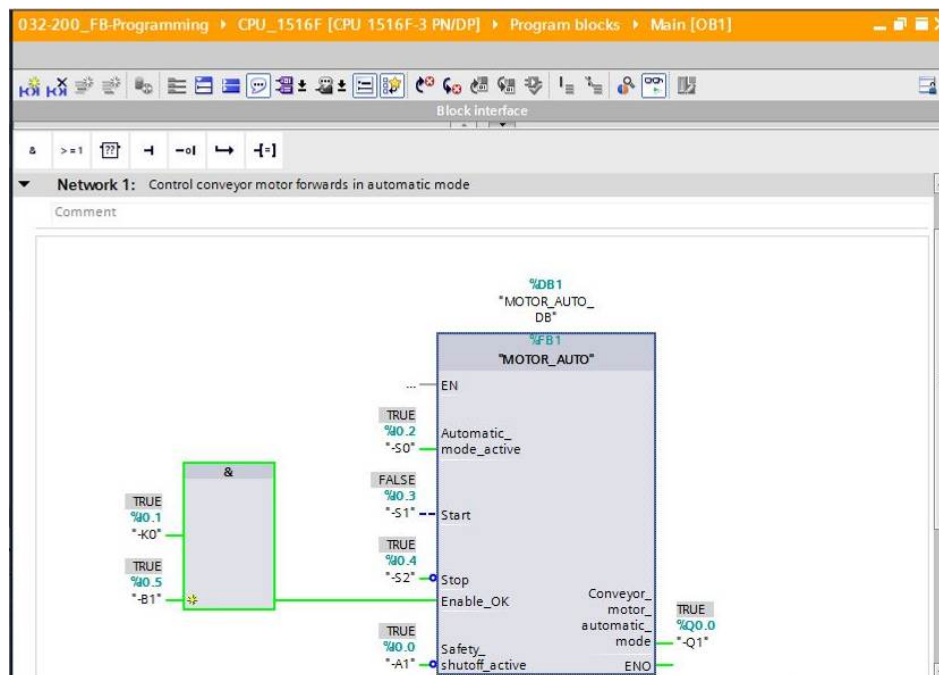
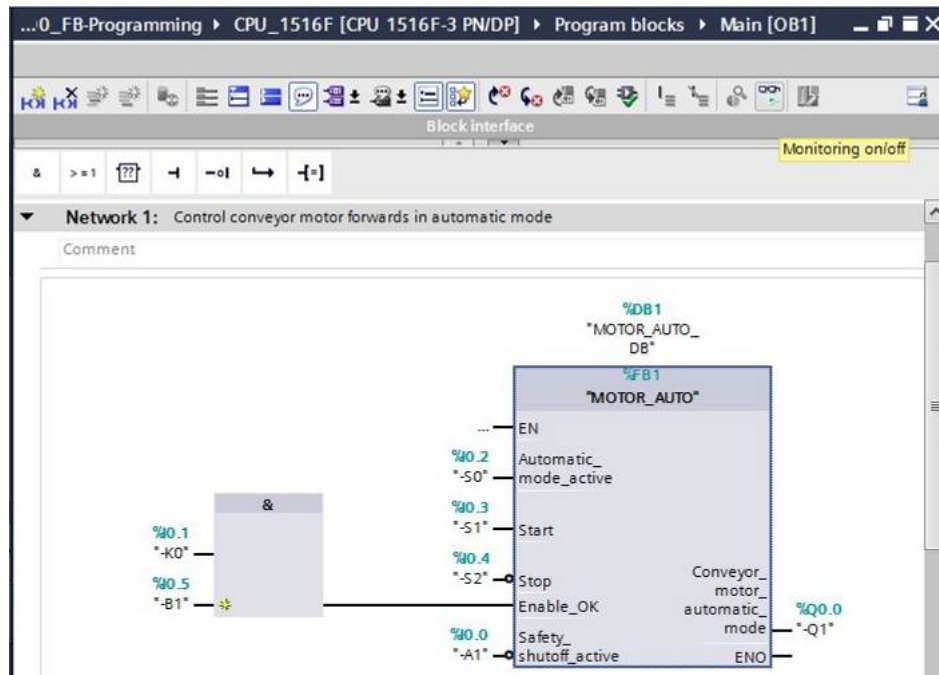
→ After successful compilation, the complete controller with the created program, as previously described in the modules for hardware configuration, can be downloaded

(→ ).



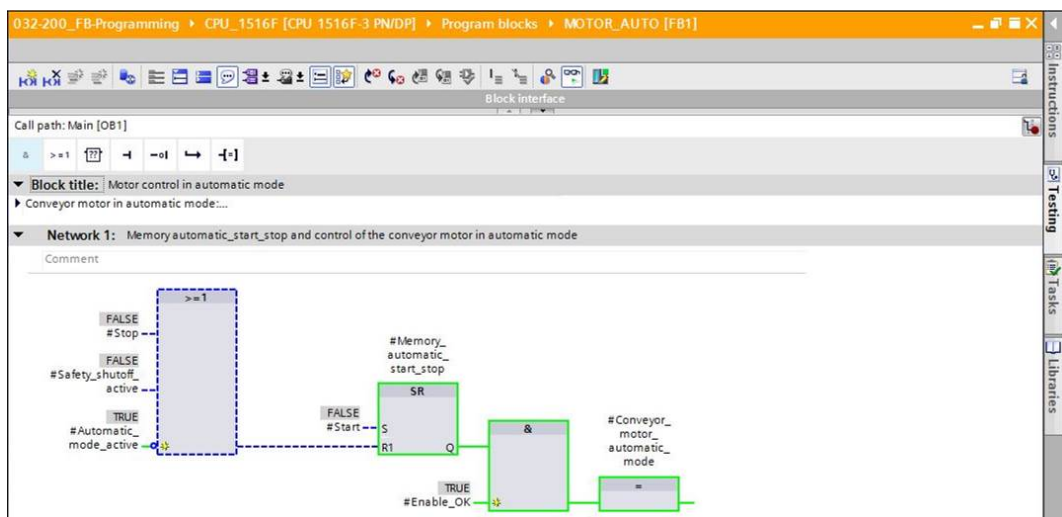
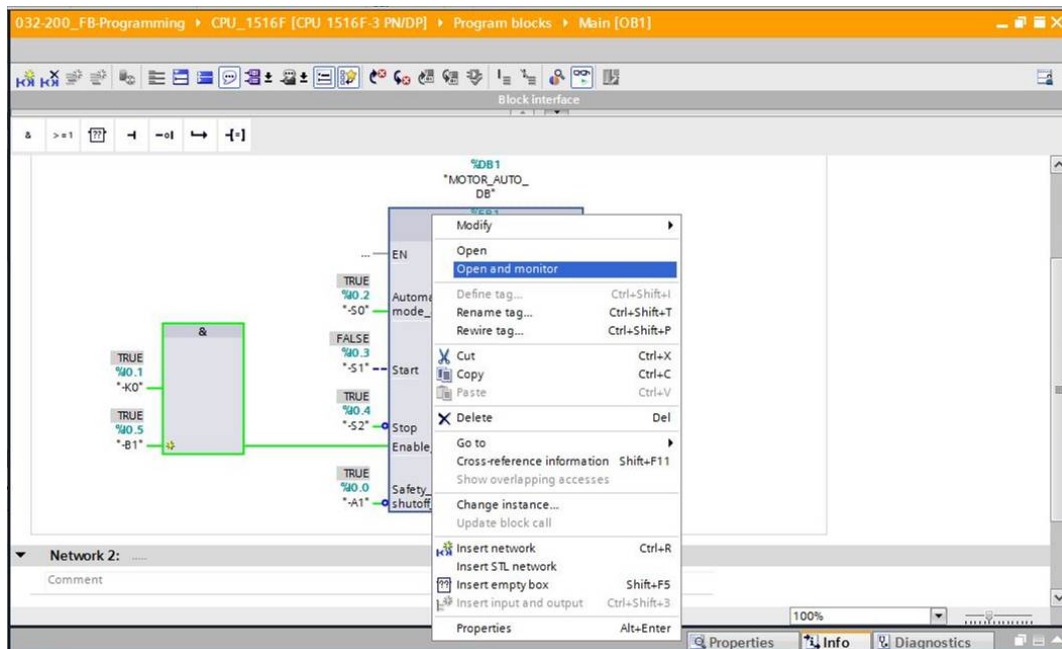
6.12 Monitor program blocks

→ The desired block must be open for monitoring the downloaded program. The monitoring can be activated/deactivated by clicking the  icon. (→ Main [OB1] → )





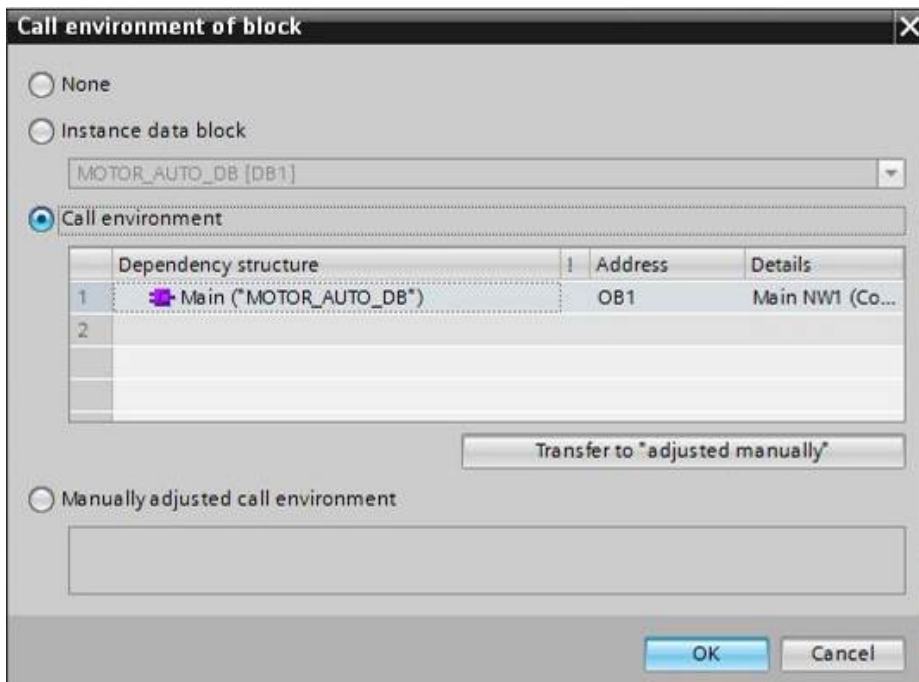
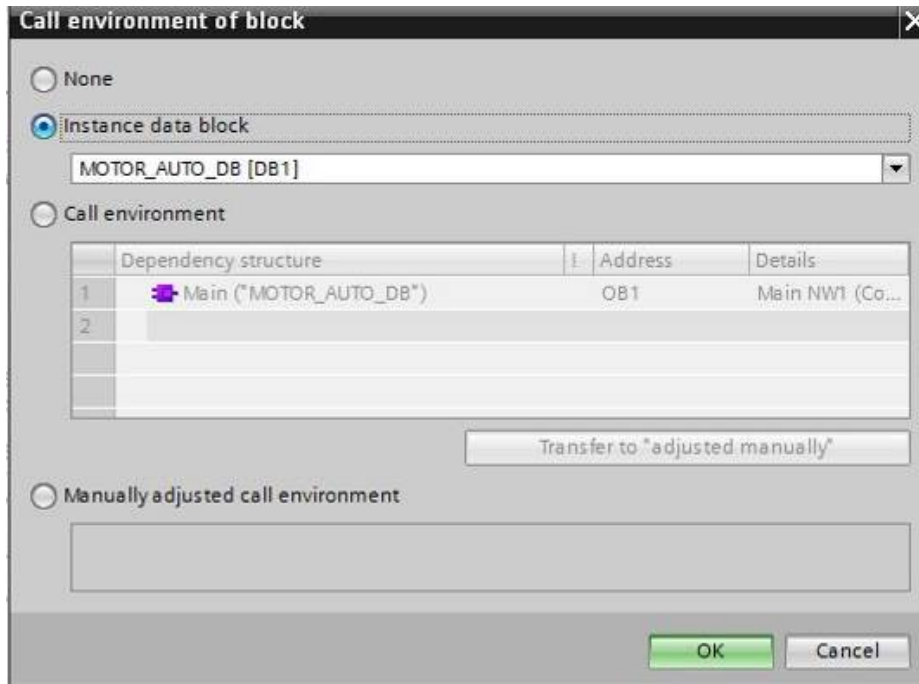
Note: The monitoring here is signal-related and controller-dependent. The signal states at the terminals are indicated with TRUE or FALSE.

→ The "MOTOR_AUTO" [FB1] function block called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking (→ "MOTOR_AUTO" [FB1] → Open and monitor).



Note: The monitoring here is function-related and controller-independent. The actuation of sensors and the station status are shown here with TRUE or FALSE.

- If a particular point of use of a "MOTOR_AUTO" [FB1] function block that is called multiple times is to be monitored, this can be performed using the  icon. There are two alternatives available for specifying the call environment: using the call environment or the instance data block. (→  → Instance data block → MOTOR_AUTO_DB1 [DB1] → Call environment → Address: OB1 → Details: Main NW1 → OK).



The screenshot displays the Siemens TIA Portal software interface. The top menu bar includes Project, Edit, View, Insert, Online, Options, Tools, Window, and Help. The left sidebar shows the Project Explorer with a tree view of the project structure, including folders for Watch and force tables, Online backups, Traces, Program info, Device proxy data, PLC alarms, Text lists, Local modules, Common data, Documentation settings, Languages & resources, Online access, and Card Reader/USB memory. The main workspace shows a ladder logic diagram for a conveyor motor control. The diagram includes a network titled "Network 1: Control conveyor motor forwards in automatic mode". The network contains a logic block labeled "MOTOR_AUTO" with inputs for "EN", "Automatic_mode_active", "Start", "Stop", "Enable_OK", "Safety_shutdown_active", "Sensor_slide", and "Sensor_end_of". The "EN" input is connected to a logic block containing an AND gate with inputs "%I0.1", "%I0.5", and "%I0.1". The "Automatic_mode_active" input is connected to "%I0.2". The "Start" input is connected to "%I0.3". The "Stop" input is connected to "%I0.4". The "Enable_OK" input is connected to "%I0.0". The "Safety_shutdown_active" input is connected to "%I0.1". The "Sensor_slide" input is connected to "%I0.1". The "Sensor_end_of" input is connected to "%I0.1". The output of the network is "Conveyor_motor_automatic_mode" connected to "%Q0.0".

7 Checklist

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) then conveyor motor forwards fixed speed (-Q1 = 1) switches on and stays on.	
4	Briefly press the automatic stop pushbutton (-S2 = 0) → Q1 = 0	
5	Activate EMERGENCY OFF (-A1 = 0) → Q1 = 0	
6	Manual mode (-S0 = 0) → Q1 = 0	
7	Switch off station (-K0 = 0) → Q1 = 0	
8	Cylinder not retracted (-B1 = 0) → Q1 = 0	
9	Project successfully archived	

8 Exercise

8.1 Task – Exercise

In this exercise, an energy saving function is to be added to the MOTOR_AUTO [FB1] function block. The expanded function block will be planned, programmed and tested:

To save energy, the conveyor should only run when a part is present.

The Conveyor_motor_automatic_mode output is therefore only activated when Memory_automatic_start_stop is set, the enable conditions are met and Memory_conveyor_start_stop is set.

The Memory_conveyor_start_stop is set when Sensor_chute_occupied signals a part and is reset when Sensor_end_of_conveyor produces a negative edge or safety shutoff is active or automatic mode is not activated (manual mode).

8.2 Planning

Plan the implementation of the task on your own.

Note: Learn about the use of the negative edge in SIMATIC S7-1500 in the online help.

8.3 Checklist – Exercise

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor at chute activated (-B4 = 1) then conveyor motor forwards fixed speed (-Q1 = 1) switches on and stays on.	
4	Sensor at end of conveyor activated (-B7 = 1) → -Q1 = 0	
5	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q1 = 0	
6	Activate EMERGENCY OFF (-A1 = 0) → -Q1 = 0	
7	Manual mode (-S0 = 0) → -Q1 = 0	
8	Switch off station (-K0 = 0) → -Q1 = 0	
9	Cylinder not retracted (-B1 = 0) → -Q1 = 0	
10	Project successfully archived	

9 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, get in touch with your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	4
2	Prerequisite.....	4
3	Theory.....	4
3.1	Instances and multi-instances in SIMATIC S7-1500	4
3.1.1	Instance data blocks/single instances	5
3.1.2	Multi-instances.....	6
4	Task	8
5	Planning.....	8
5.1	Automatic mode - Conveyor motor with time function.....	8
6	Structured step-by-step instructions	9
6.1	Retrieve an existing project	9
6.2	Addition of an IEC timer TP to function block FB1 "MOTOR_AUTO"	11
6.3	Update the block call in the organization block	17
6.4	Save and compile the program.....	18
6.5	Download the program	19
6.6	Monitor program blocks	20
6.7	Archive the project.....	22
7	Checklist	23
8	Exercise	24
8.1	Task – Exercise	24
8.2	Planning.....	24
8.3	Checklist – Exercise	25
9	Additional information	26

IEC TIMERS AND IEC COUNTERS MULTI-INSTANCES FOR SIMATIC S7-1500

1 Goal

In this chapter, you will become acquainted with the use of single instances and multi-instances for programming of the SIMATIC S7-1500 with the TIA Portal programming tool.

The module explains the various types of instance data blocks and shows step-by-step how to add IEC timers and IEC counters to a program block.

2 Prerequisite

This chapter builds on the FB programming with the SIMATIC S7 CPU1516F-3 PN/DP. You can use the following project for this chapter, for example: 032-200_FBProgramming_R1503.zap13

3 Theory

3.1 Instances and multi-instances in SIMATIC S7-1500

The call of a function block is referred to as an **instance**. An **instance** is assigned to every call of a function block and serves as a data memory. It stores the actual parameters and the static data of the function block.

The tags declared in the function block determine the structure of the instance data block.

Use of single instances and multi-instances

You can assign instances as follows:

Call as a **single instance**:

- A separate instance data block for each instance of a function block

Call as a **multi-instance**:

- One instance data block for several instances of one or more function blocks

3.1.1 Instance data blocks/single instances

The call of a function block that is assigned its own instance data block is called a **single instance**.

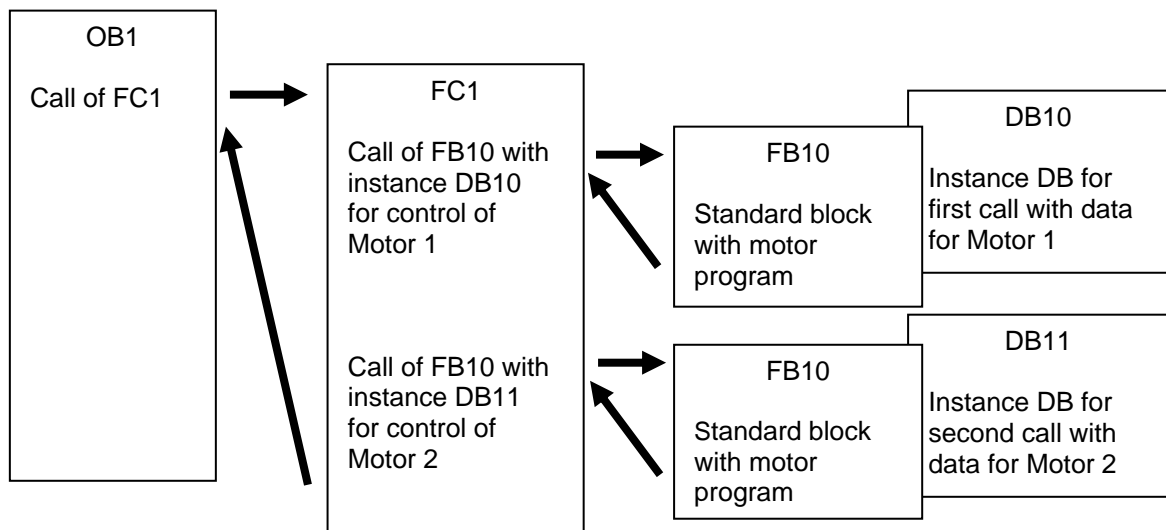
If the function block was created according to the rules for library-compatible standard blocks, it can also be called multiple times.

However, you must assign another instance data block for each call as a single instance.

Example of single instances:

The following figure shows the control of two motors using one function block FB10 and two different data blocks:

The different data for the individual motors, such as speed, acceleration time and total operating time, are saved in the instance data blocks DB10 and DB11.



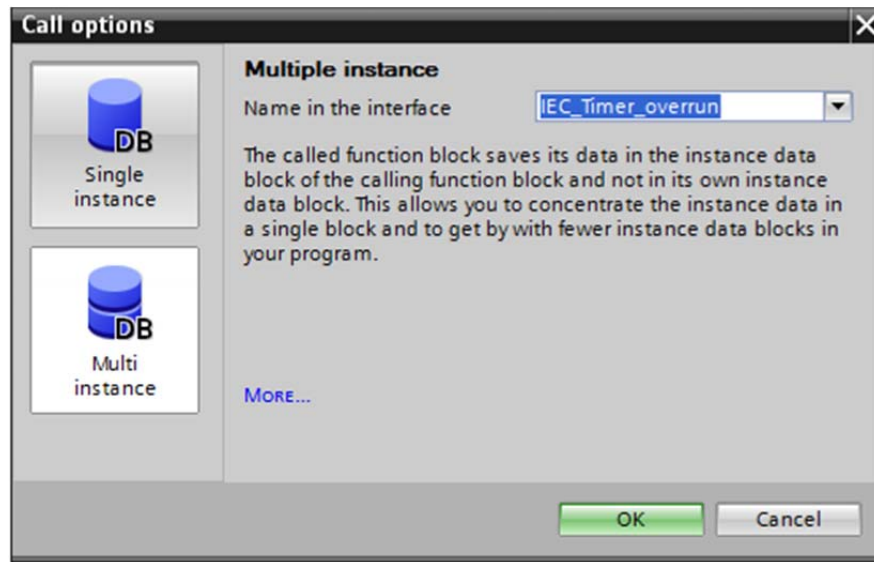
Note: Some commands, such as timers and counters, react like function blocks. When these are called, they also require an assigned memory area, e.g. in the form of an instance data block.

3.1.2 Multi-instances

You may want to limit the number of data blocks used for instances or this may be necessary due to lack of memory in the utilized CPU.

If other function blocks, timers, counters, etc. that already exist will be called in a function block in your user program, you can call these other function blocks without separate (i.e., additional) instance DBs.

Simply select '**Multi-instance**' for the call options:



Notes: Multi-instances enable a called function block to store its data in the instance data block of the calling function block.

In this case, the calling block must always be a function block.

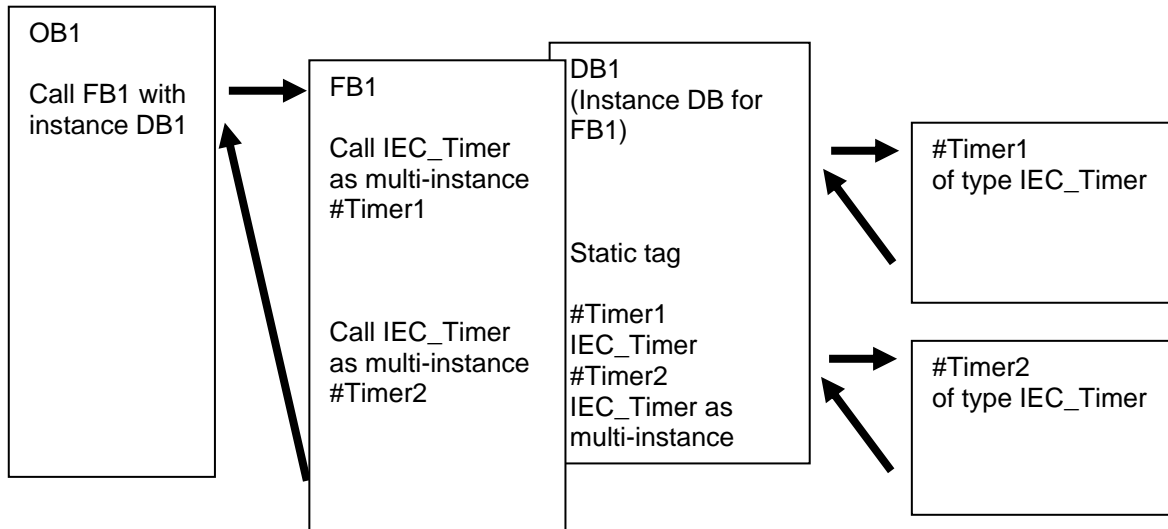
This allows you to concentrate the instance data in one instance data block and thus make better use of the number of DBs available.

Incidentally, this is always required when the calling block is to remain available for reuse as a standard block.

Example of multi-instances:

The following figure shows two calls of an IEC_Timer of type TP (pulse) within a function block.

The different data for the two counters is stored as different **multi-instances** in the instance data block DB1 of the calling function block FB1.



4 Task

In this chapter, an IEC timer will be added to the function block from chapter "SCE_EN_032-200 FB Programming".

5 Planning

The IEC timer is programmed as an addition to the MOTOR_AUTO [FB1] function block from the "032-200_FBProgramming.zap13" project. This project must be retrieved in order to add the IEC timer TP (latching pulse). A multi-instance will be created as a memory for the timer.

5.1 Automatic mode - Conveyor motor with time function

The Memory_automatic_start_stop is latched with Start but only if the reset conditions are not present.

The Memory_automatic_start_stop is reset if Stop is present or safety shutoff is active or automatic mode is not activated (manual mode).

The Conveyor_motor_automatic_mode output is activated when Memory_automatic_start_stop is set, the enable conditions are met and Memory_conveyor_start_stop is set.

To save energy, the conveyor should only run when a part is present.

For this reason, the Memory_conveyor_start_stop is set when Sensor_chute_occupied signals a part and reset when Sensor_end_of_conveyor produces a negative edge or safety shutoff is active or automatic mode is not activated (manual mode).

Addition of time function:

Because the Sensor_end_of_conveyor is not able to be mounted directly at the end of the conveyor, the Sensor_end_of_conveyor signal must be stretched.

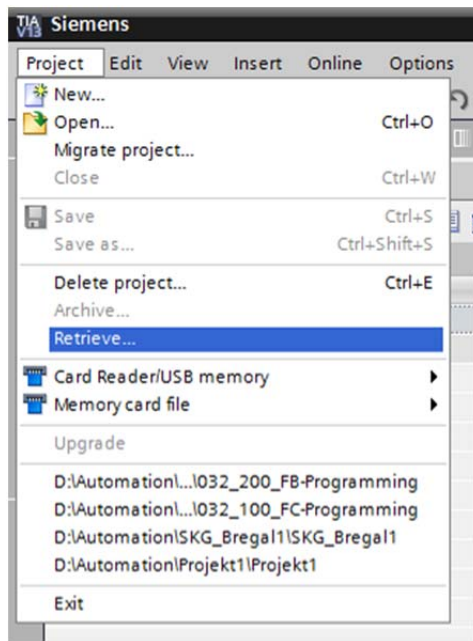
To achieve this, a latching pulse will be inserted between Sensor_end_of_conveyor and the negative edge detection.

6 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

6.1 Retrieve an existing project

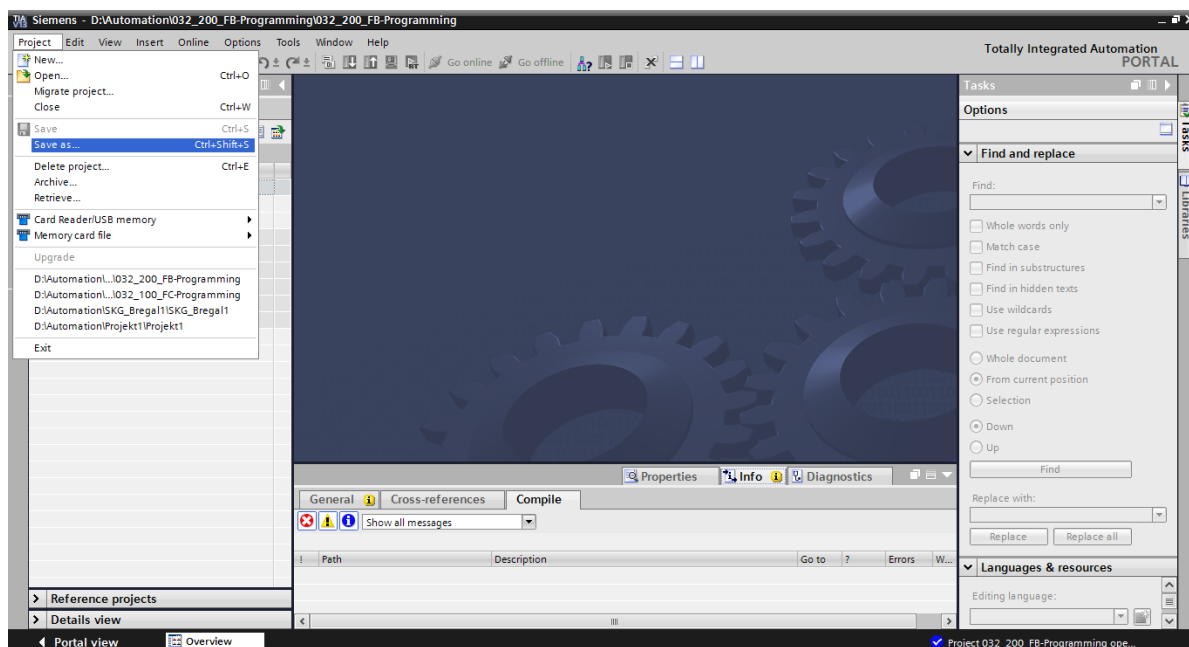
- Before we can expand the "MOTOR_AUTO [FB1]" function block, we must retrieve the "032-200_FBProgramming.zap13" project from chapter "SCE_EN_032-200 FBProgramming". To retrieve an existing project that has been archived, you must select the relevant archive with
- Project → Retrieve in the project view. Confirm your selection with Open. (→ Project → Retrieve → Select a .zap archive → Open)



- The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK". (→ Target directory → OK)

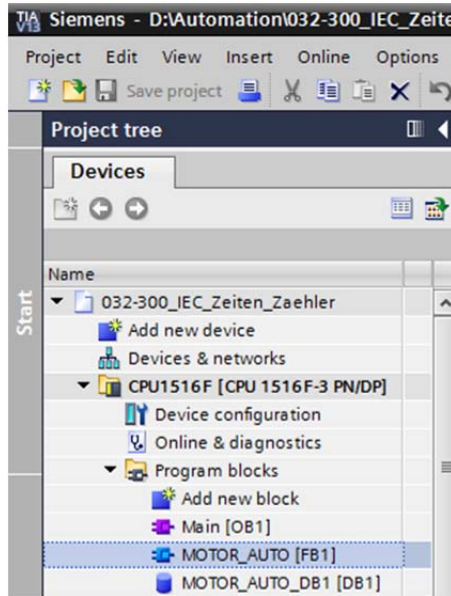
Save the opened project under the name 032-300_IEC_Timers_Counters.


→ Project → Save as ... → 032-300-IEC_Timers_Counters → Save)

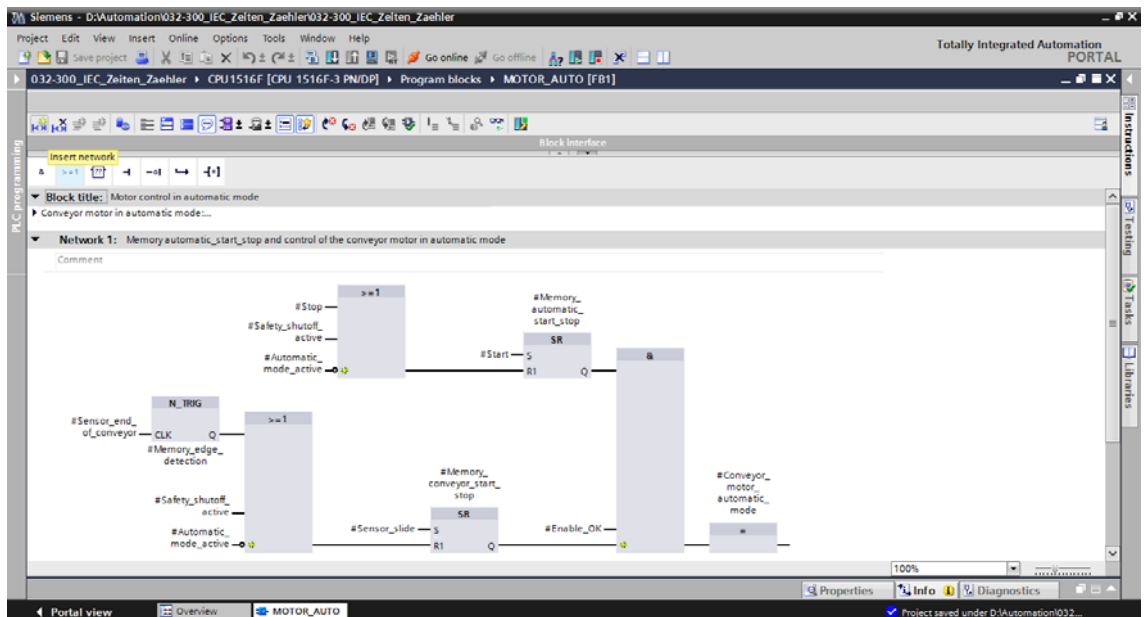


6.2 Addition of an IEC timer TP to function block FB1 "MOTOR_AUTO"

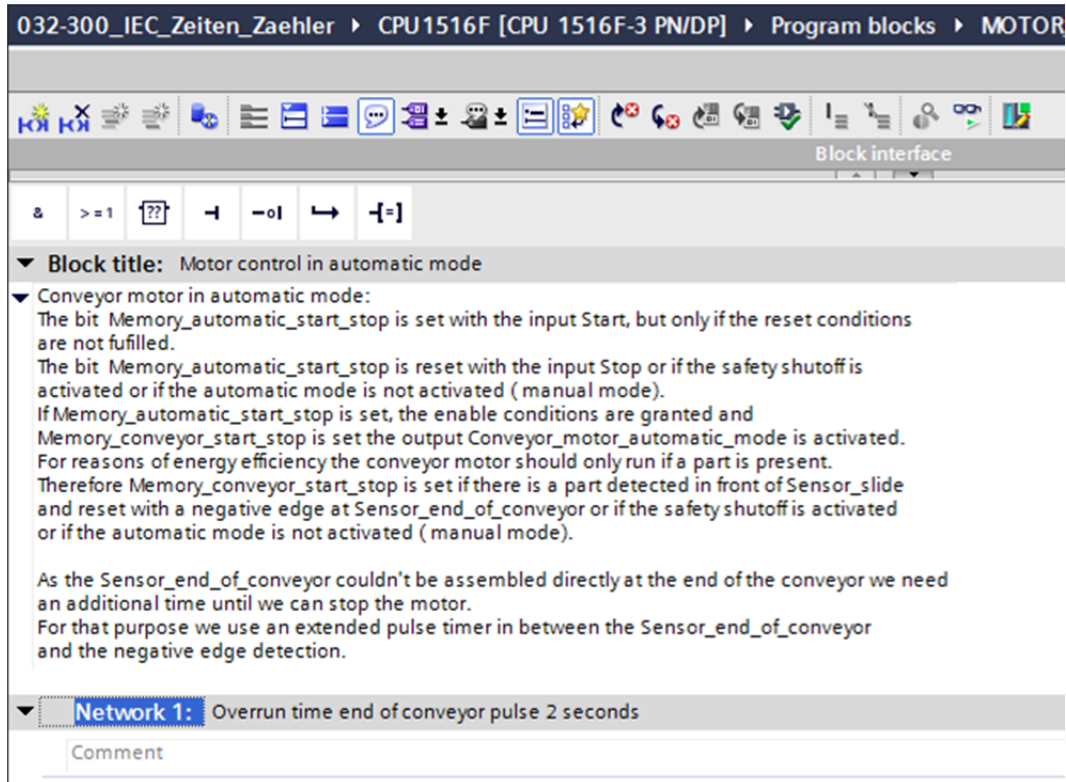
→ First, open the "MOTOR_AUTO [FB1]" function block with a double-click.



→ Insert another network at the beginning of the "MOTOR_AUTO [FB1]" function block by selecting the → "block title" and then clicking the →  icon for "Insert network".

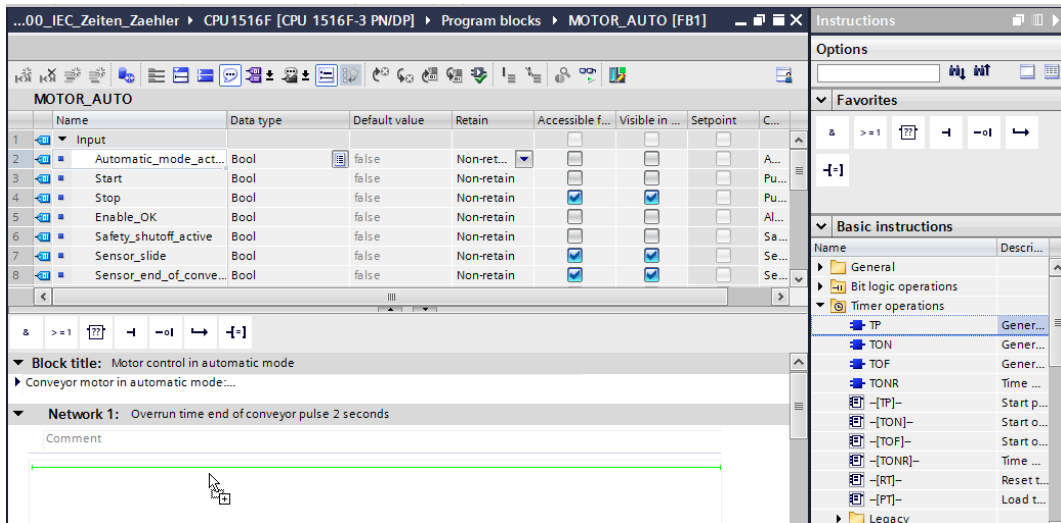


→ Add helpful information to the block comment and the network title of "Network 1:".

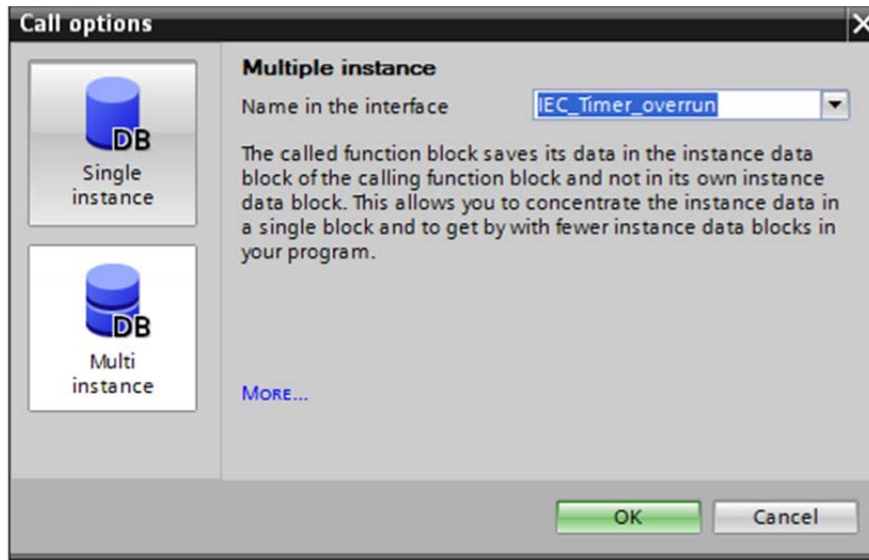


→ On the right side of your programming window, you will see the timer functions in the list of instructions. Under → Basic instructions → Timer operations, find function **TP** (Generate pulse) and use a drag-and-drop operation to move it to Network 1 (green line appears, mouse pointer with + symbol).

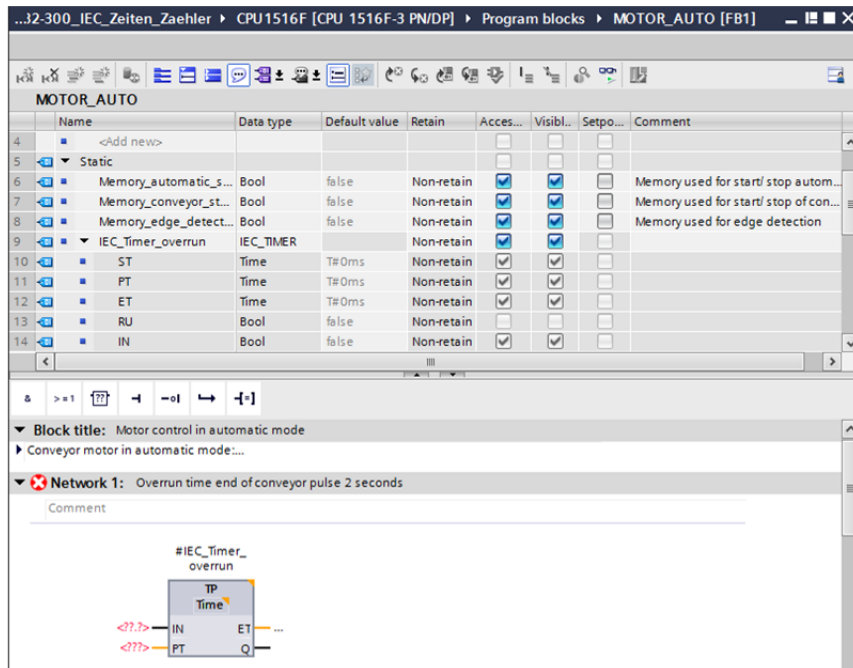
(→ Instructions → Basic instructions → Timer operations → **TP**)





- The timer function requires a memory. Here, this memory is made available within the instance data block by the function block without the creation of a new instance data block. Select the → "Multi-instance" option for this. Enter a name for the multi-instance and confirm with → "OK". (→ Multi-instance → IEC_Timer_overrun → OK)

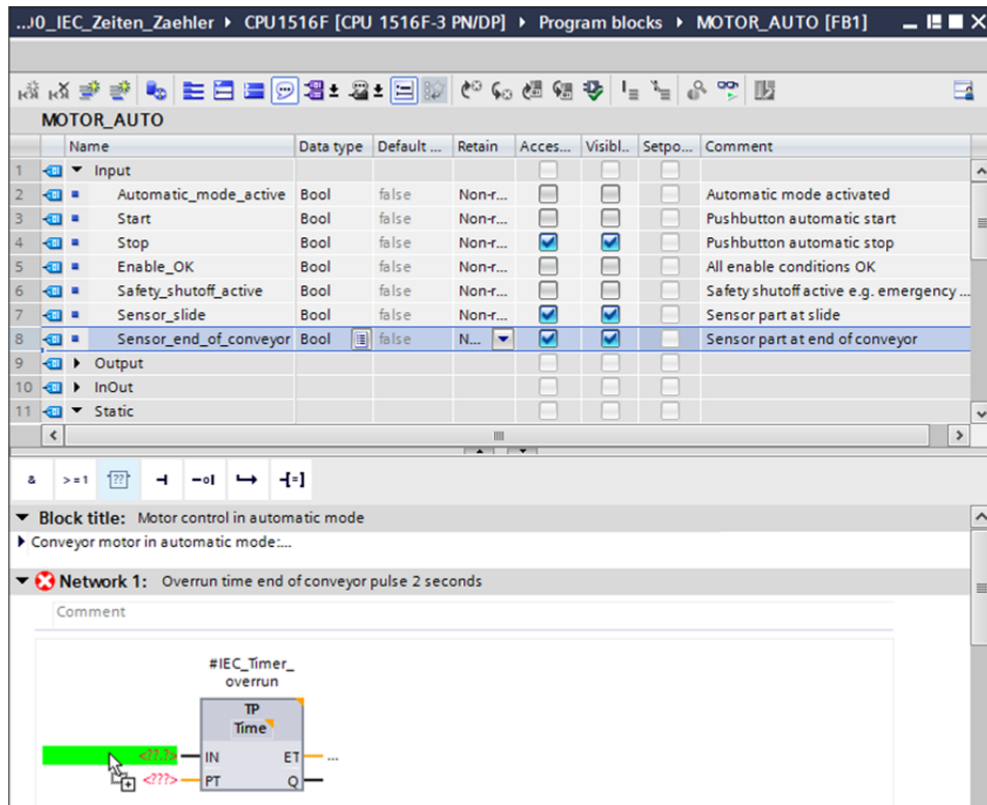


- As a result, a tag structure of "Static" type suitable for TP Timer will be created in the interface description.

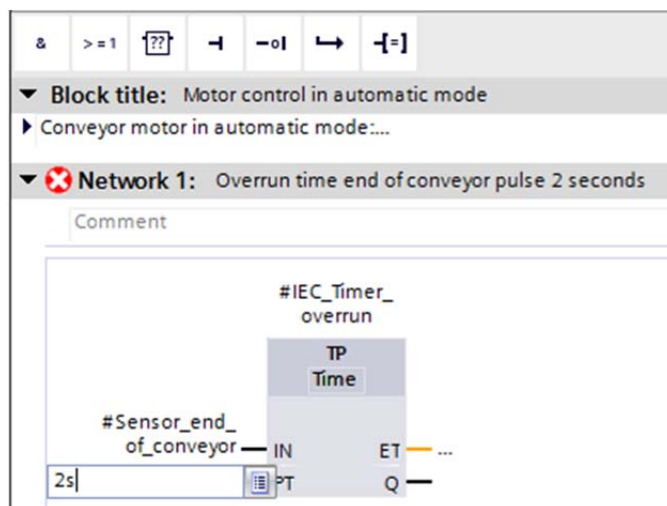


Note: A multi-instance can only be used for programming within a function block because static tags are only available there.

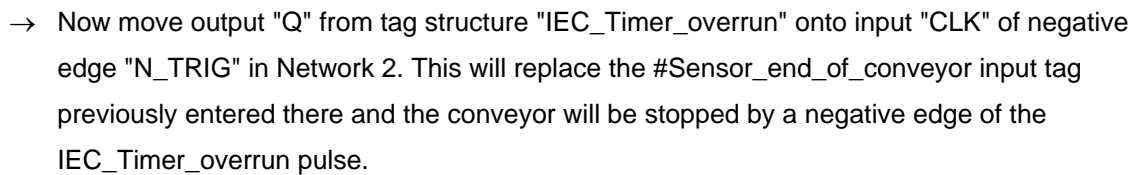
- Use drag-and-drop to move input parameter #Sensor_end_of_conveyor to **<??.?>** in front of parameter "IN" of TP Timer so that this will be started at a positive edge at input #Sensor_end_of_conveyor. The best way to select a parameter in the interface description is by "grabbing" it at the blue symbol.  (→  Sensor_end_of_conveyor)



- Enter the required pulse duration of 2 seconds in front of parameter "PT" (→ 2s)



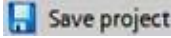
8



The screenshot displays the Siemens STEP 7 HW Config software interface. The top window shows the 'MOTOR_AUTO' hardware configuration table, which lists various digital inputs and outputs for a motor control system. The table includes columns for Name, Data type, Default value, Retain status, Access status, Visibility, Setpoint, and Comment.

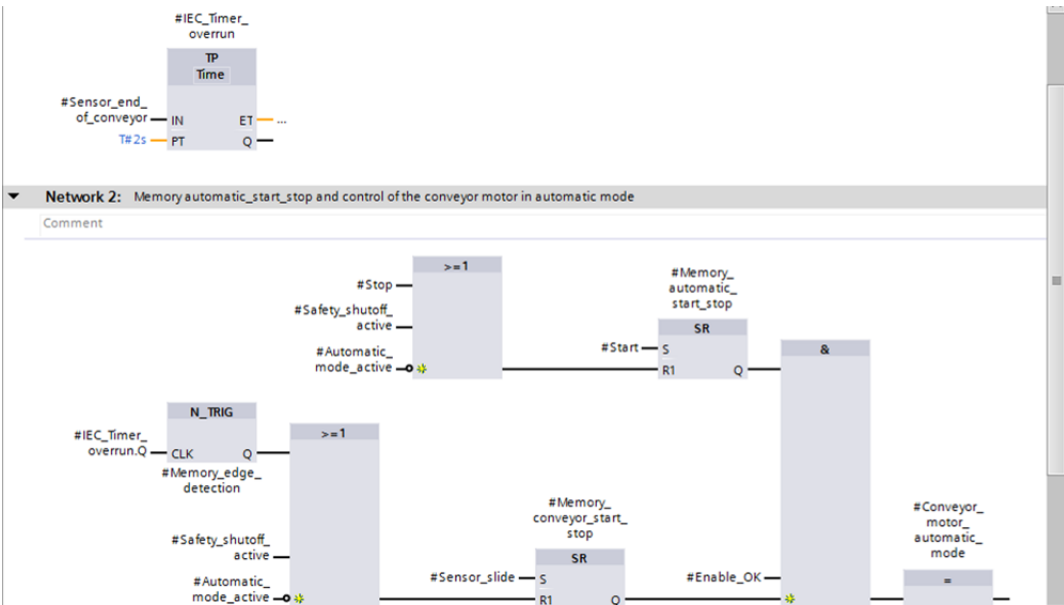
	Name	Data type	Default ...	Retain	Acces...	Visibl...	Setpo...	Comment
4	Static							
5	Memory_automatic_sta...	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Memory used for start/ stop automatic mode
6	Memory_conveyor_start...	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Memory used for start/ stop of conveyor in aut...
7	Memory_edge_detection	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Memory used for edge detection
8	IEC_Timer_overrun	IEC_TIMER		Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
9	ST	Time	T#0ms	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
10	PT	Time	T#0ms	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
11	ET	Time	T#0ms	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
12	RU	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
13	IN	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
14	Q	Bool	false	N...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
15	Temp							

Below the table, a ladder logic diagram is visible, showing the interconnection of various digital signals and timers. The diagram includes inputs like #Stop, #Safety_shutoff_active, #Automatic_mode_active, #Sensor_end_of_conveyor, #Memory_edge_detection, #Safety_shutoff_active, #Automatic_mode_active, #Sensor_slide, and #Enable_OK. It also shows outputs like #Memory_automatic_start_stop, #Memory_conveyor_start_stop, and #Conveyor_motor_automatic_mode. The logic involves set-reset (SR) and set (S) coils, as well as AND and OR gates.

→ Do not forget to click  Save project. The finished function block "MOTOR_AUTO" [FB1] with the timer is shown in FBD below.

032-300_IEC_zeiten_Zaehler ▸ CPU1516F [CPU 1516F-3 PN/DP] ▸ Program blocks ▸ MOTOR_AUTO [FB1]

Name	Data type	Default ...	Retain	Access...	Visibl...	Setpo...	Comment
1 ▾ Input							
2 Automatic_mode_active	Bool	false	Non-r...				Automatic mode activated
3 Start	Bool	false	Non-r...				Pushbutton automatic start
4 Stop	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Pushbutton automatic stop
5 Enable_OK	Bool	false	Non-r...				All enable conditions OK
6 Safety_shutoff_active	Bool	false	Non-r...				Safety shutoff active e.g. emergency stop o...
7 Sensor_slide	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Sensor part at slide
8 Sensor_end_of_conveyor	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Sensor part at end of conveyor
9 ▾ Output							
10 Conveyor_motor_auto...	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Control of the conveyor motor in automati...
11 ▾ InOut							
12 ▾ Static							
13 Memory_automatic_sta...	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Memory used for start/ stop automatic mode
14 Memory_conveyor_start...	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Memory used for start/ stop of conveyor in ...
15 Memory_edge_detection	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Memory used for edge detection
16 ▾ IEC_Timer_overrun	IEC_TIMER		Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
17 ST	Time	T#0ms	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
18 PT	Time	T#0ms	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
19 ET	Time	T#0ms	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
20 RU	Bool	false	Non-r...				
21 IN	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
22 Q	Bool	false	N...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
23 ▾ Temp							

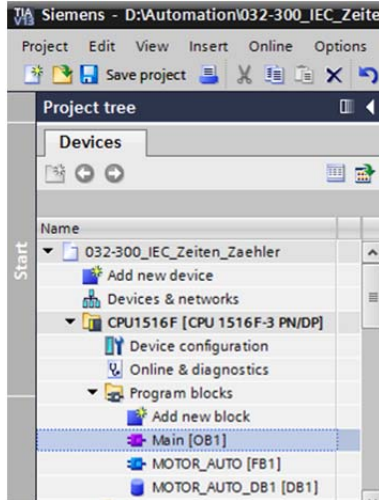




Network 2: Memory automatic_start_stop and control of the conveyor motor in automatic mode

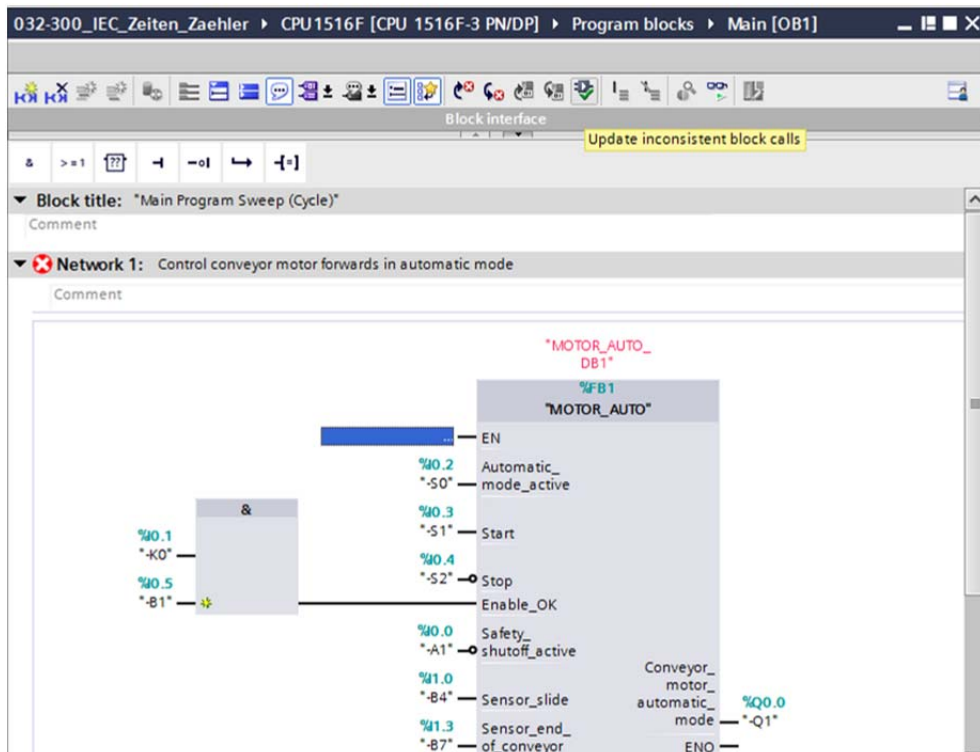
Comment

6.3 Update the block call in the organization block





→ Open the "Main [OB1]" organization block with a double-click.

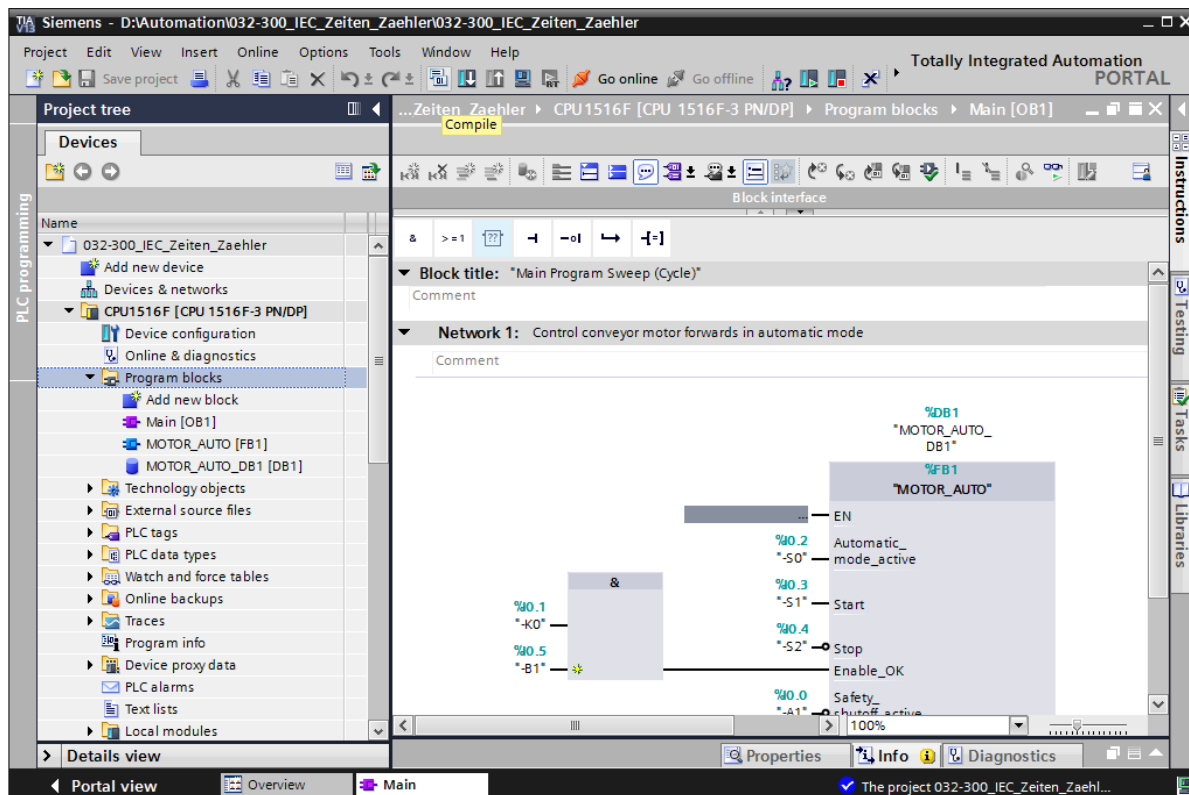


→ In Network 1 of the "Main [OB1]" organization block, instance data block "MOTOR_AUTO_DB1" for the "MOTOR_AUTO [FB1]" function block appears incorrect, because the additional memory for the TP Timer has not yet been added there. Click the → "  " icon for "Update inconsistent block calls". This will add the "MOTOR_AUTO_DB1" instance data block correctly again. (→ )



6.4 Save and compile the program


- To save your project, select the  **Save project** button in the menu. To compile all blocks, click the "Program blocks" folder and select the  icon for compiling in the menu (→  **Save project** → Program blocks → ).

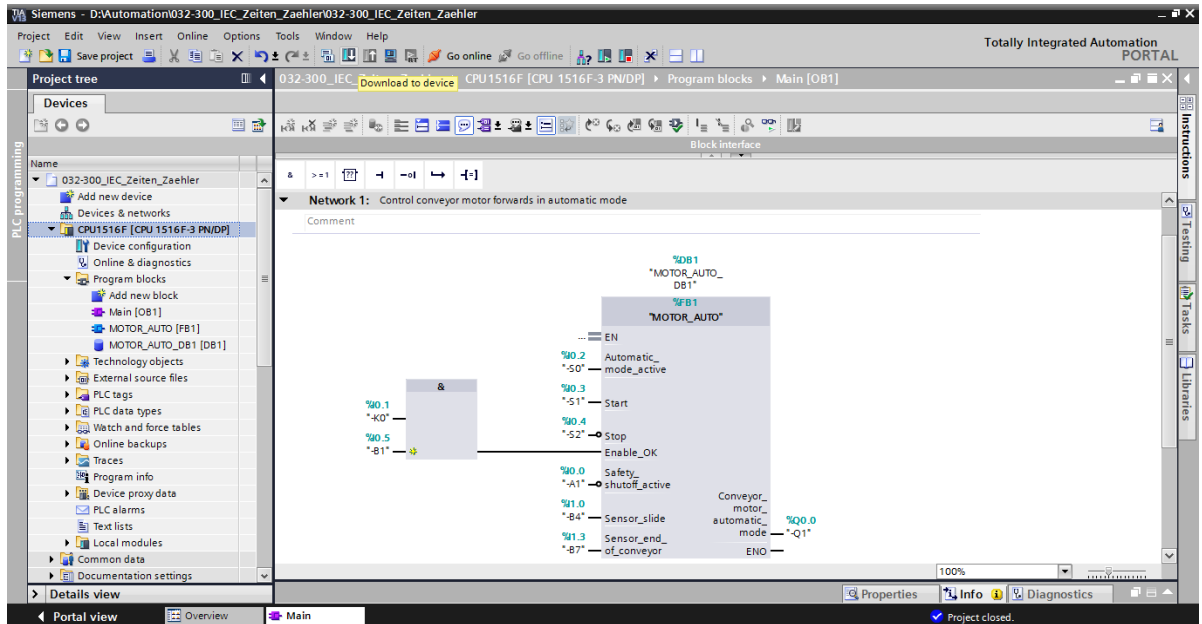


- The "Info", "Compile" area shows which blocks were successfully compiled.

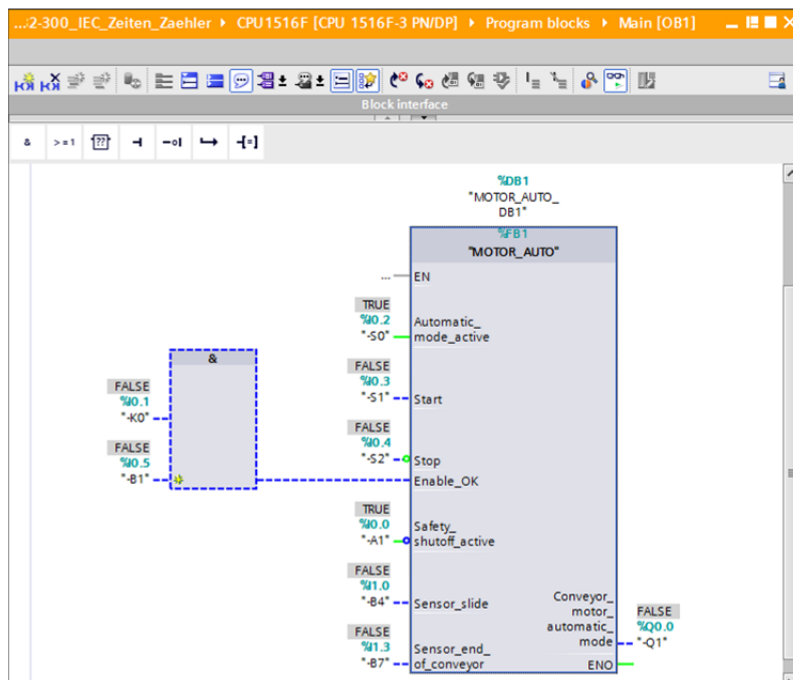
Properties Info Diagnostics							
General Cross-references Compile Syntax							
Show all messages							
Compiling completed (errors: 0; warnings: 0)							
Path	Description	Go to	?	Errors	Warnings	Time	
▼ CPU1516F				0	0	6:47:14 AM	
▼ Program blocks				0	0	6:47:14 AM	
MOTOR_AUTO (FB1)	Block was successfully compiled.					6:47:14 AM	
Main (OB1)	Block was successfully compiled.					6:47:14 AM	
	Compiling completed (errors: 0; warnings: 0)					6:47:14 AM	

6.5 Download the program

→ After successful compilation, the complete controller with the created program including the hardware configuration, as previously described in the modules, can be downloaded. (→ )



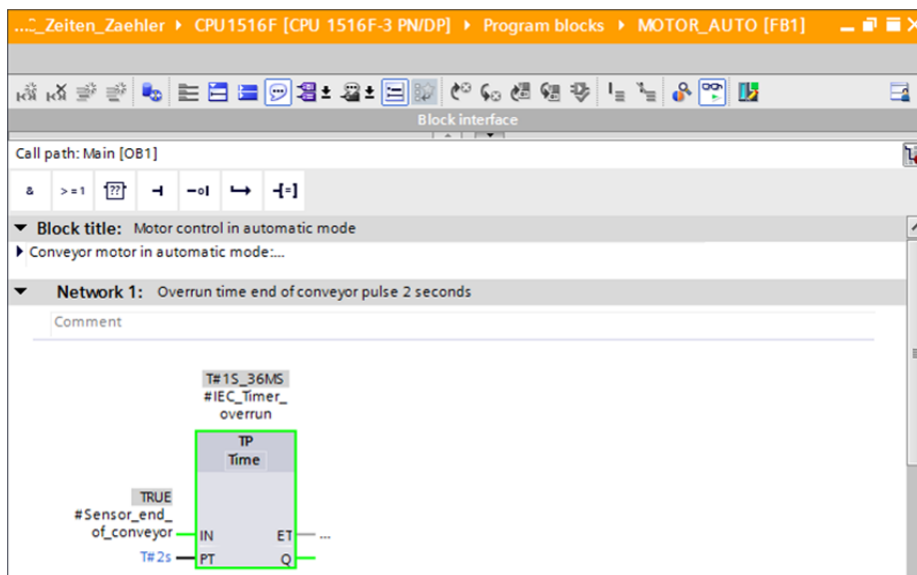
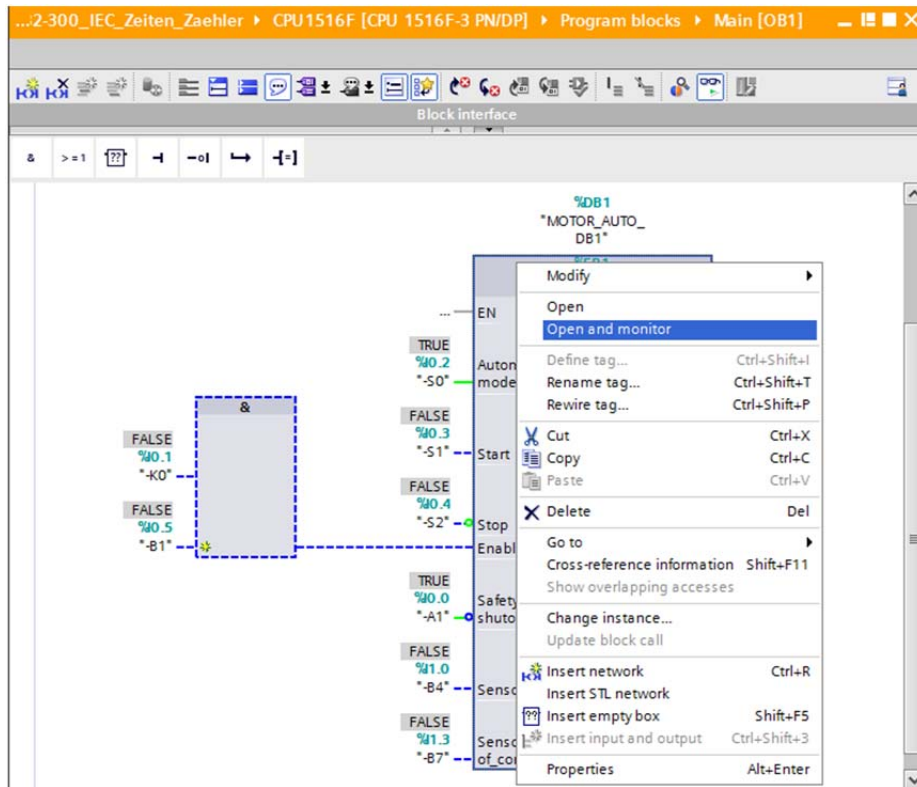
8



Note: The monitoring here is signal-related and controller-dependent. The signal states at the terminals are indicated with TRUE or FALSE.

→ The "MOTOR_AUTO" [FB1] function block called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking, thereby allowing the program code in the function block with the TP Timer to be monitored.

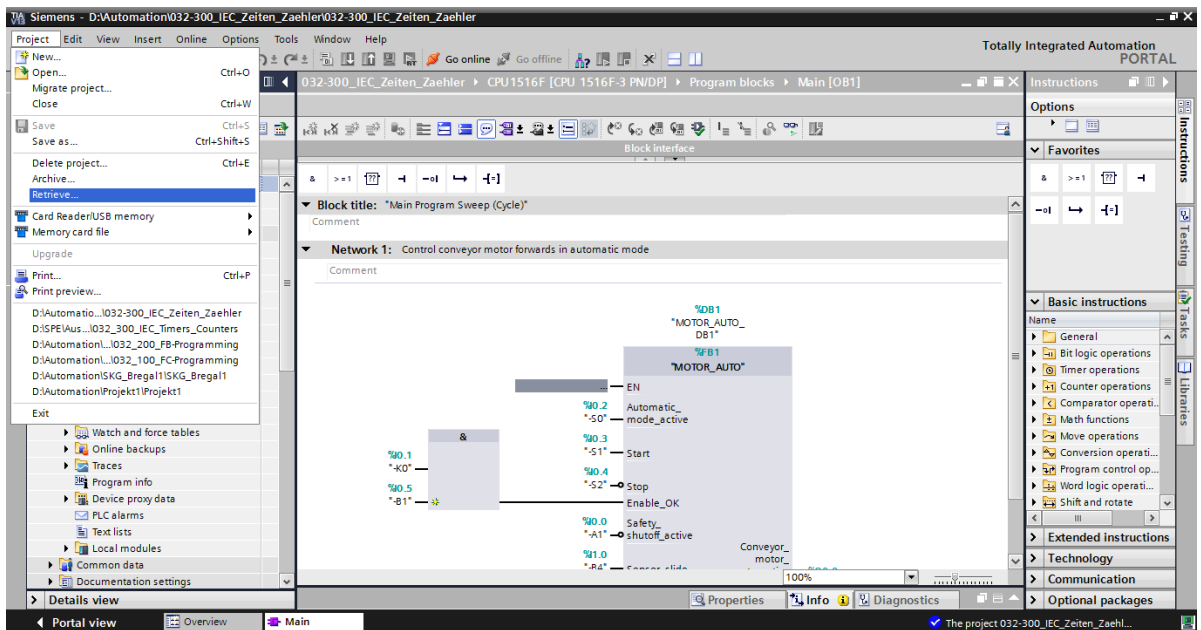
(→ "MOTOR_AUTO" [FB1] → Open and monitor)



Note: The monitoring here is function-related and controller-independent. The actuation of sensors and the station status are shown here with TRUE or FALSE.

6.7 Archive the project

- As the final step, we want to archive the complete project. Select the → "Archive ..." command in the → "Project" menu. Select a folder where you want to archive your project and save it with the file type "TIA Portal project archive". (→ Project → Archive → TIA Portal project archive → 032-300_IEC_Timers_Counters.... → Save)



7 Checklist

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor at chute activated (-B4 = 1) then conveyor motor forwards fixed speed (-Q1 = 1) switches on and stays on.	
4	Sensor at end of conveyor activated (-B7 = 1) → -Q1 = 0 (after 2 seconds)	
5	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q1 = 0	
6	Activate EMERGENCY OFF (-A1 = 0) → -Q1 = 0	
7	Manual mode (-S0 = 0) → -Q1 = 0	
8	Switch off station (-K0 = 0) → -Q1 = 0	
9	Cylinder not retracted (-B1 = 0) → -Q1 = 0	
10	Project successfully archived	

8 Exercise

8.1 Task – Exercise

In this exercise, an IEC counter is also to be added to the MOTOR_AUTO [FB1] function block. The expanded function block will be planned, programmed and tested:

The magazine for plastic holds only 5 parts and the parts will therefore be counted at the end of the conveyor.

When 5 parts are stored in the magazine, automatic mode is to be stopped.

Once the magazine has been emptied, automatic mode will be restarted with

Start_command is started again and the counter is reset.

8.2 Planning

Plan the implementation of the task on your own.

Note: Learn about the use of IEC counters in SIMATIC S7-1500 in the online help.

8.3 Checklist – Exercise

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (S1 = 1) Sensor at chute activated (-B4 = 1) then conveyor motor forwards fixed speed (-Q1 = 1) switches on and stays on.	
4	Sensor at end of conveyor activated (-B7 = 1) → -Q1 = 0 (after 2 seconds)	
5	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q1 = 0	
6	Activate EMERGENCY OFF (-A1 = 0) → -Q1 = 0	
7	Manual mode (-S0 = 0) → -Q1 = 0	
8	Switch off station (-K0 = 0) → -Q1 = 0	
9	Cylinder not retracted (-B1 = 0) → -Q1 = 0	
10	5th part in magazine → -Q1 = 0	
11	Project successfully archived	

9 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, please contact your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

9

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	4
2	Prerequisite.....	4
3	Theory.....	4
3.1	Fault diagnostics and hardware faults	4
3.2	Hardware diagnostics	5
3.3	Diagnostics for program blocks	6
4	Task	7
5	Planning.....	7
5.1	Online interface.....	7
6	Structured step-by-step instructions	8
6.1	Retrieve an existing project	8
6.2	Download the program	9
6.3	Go online	11
6.4	Online & diagnostics of the SIMATIC S7 controller.....	15
6.5	Online/offline comparison	24
6.6	Monitor and modify tags	27
6.7	Force tags.....	30
6.8	Checklist	34
7	Exercise	35
7.1	Task – Exercise	35
7.2	Planning.....	35
7.3	Checklist – Exercise	36
8	Additional information	37

BASICS OF DIAGNOSTIC FUNCTIONS

1 Goal

In this module, the reader will become acquainted with the tools that support troubleshooting.

This module will present diagnostic functions that, for example, you can test with the TIA project from the SCE_EN_032-100_FC-Programming with SIMATIC S7-1500 module.

2 Prerequisite

This chapter builds on the hardware configuration of SIMATIC S7 CPU1516F-3 PN/DP. However, other hardware configurations that have digital input and output cards can be used. You can use the following project for this chapter, for example:

SCE_EN_032_100_FC-Programming_R1503.zap13

3 Theory

3.1 Fault diagnostics and hardware faults

Faults can be caused by a variety of things.

Two error patterns can be distinguished for faults that occur after a transition to RUN.

1. The CPU goes to or stays in STOP mode. The yellow STOP LED lights up, and other indicator LEDs light up on the CPU, power supply unit, I/O modules or bus modules.

A CPU fault has occurred in this case. For example, a module in the AS might be defective or have an incorrect parameter assignment, or a bus system fault might be present.

An interruption analysis will be performed in this instance by evaluating the hardware diagnostics and by reading the module status in the diagnostic buffer of the CPU.

2. The CPU is in faulty RUN mode. The green RUN LED lights up and other indicator LEDs light up or flash on the CPU, power supply unit, I/O modules or bus modules.

In this case, a fault may have occurred in the I/O devices or power supply.

A visual check is performed initially to narrow down the fault area. The indicator LEDs on the CPU and I/O devices are evaluated. The diagnostic data of the faulty I/O and bus modules are read from the hardware diagnostics. In addition, a fault analysis can be performed using a watch table on the programming device.

3.2 Hardware diagnostics

The device view in online mode of the TIA Portal provides you a quick overview of the configuration and system status of the automation system.

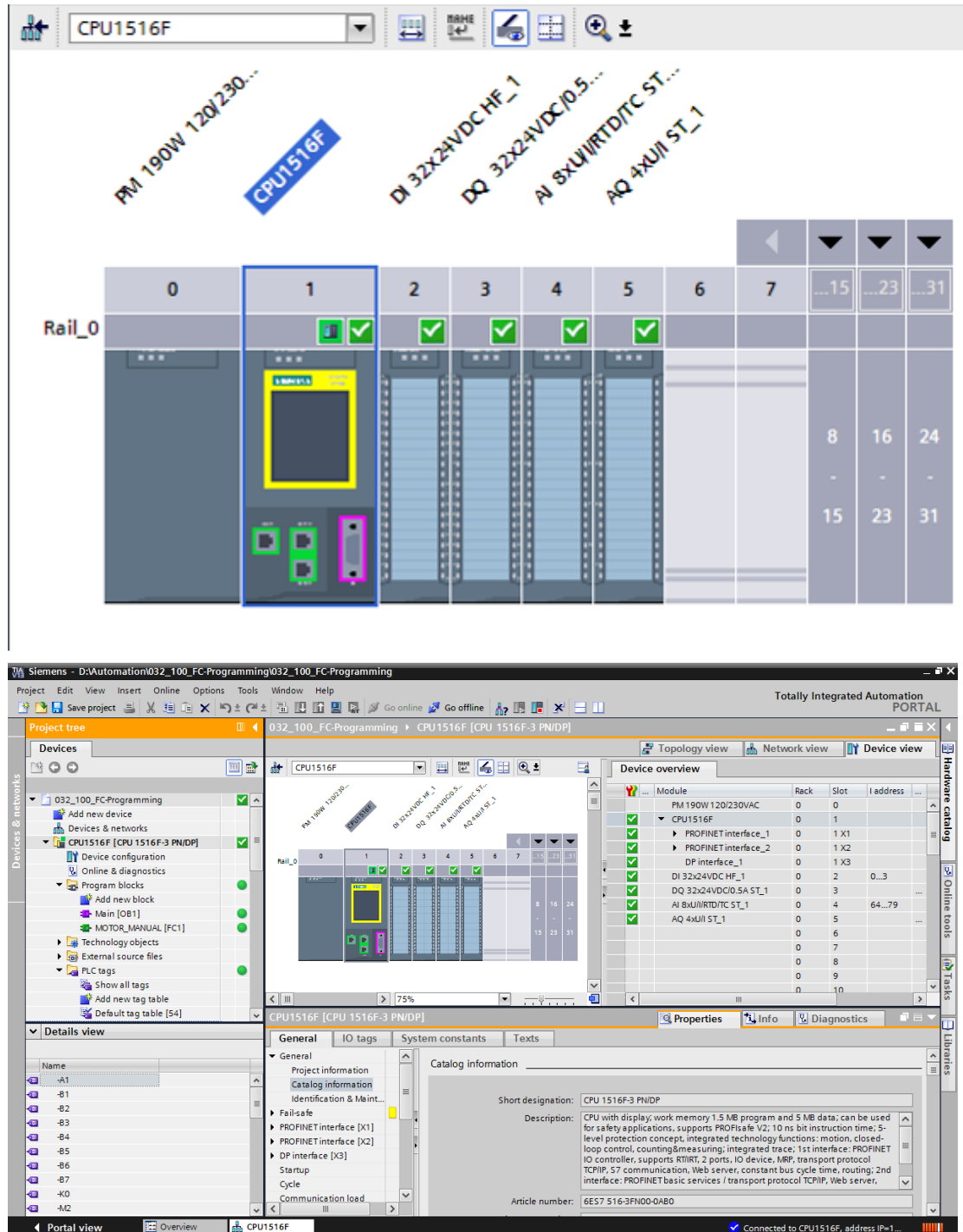


Figure 1: Online view of device configuration

3.3 Diagnostics for program blocks

The project tree window of the TIA Portal in online mode provides you an overview of the programmed blocks of the user program. With the help diagnostic symbols, a comparison of the program blocks used offline and online is displayed.

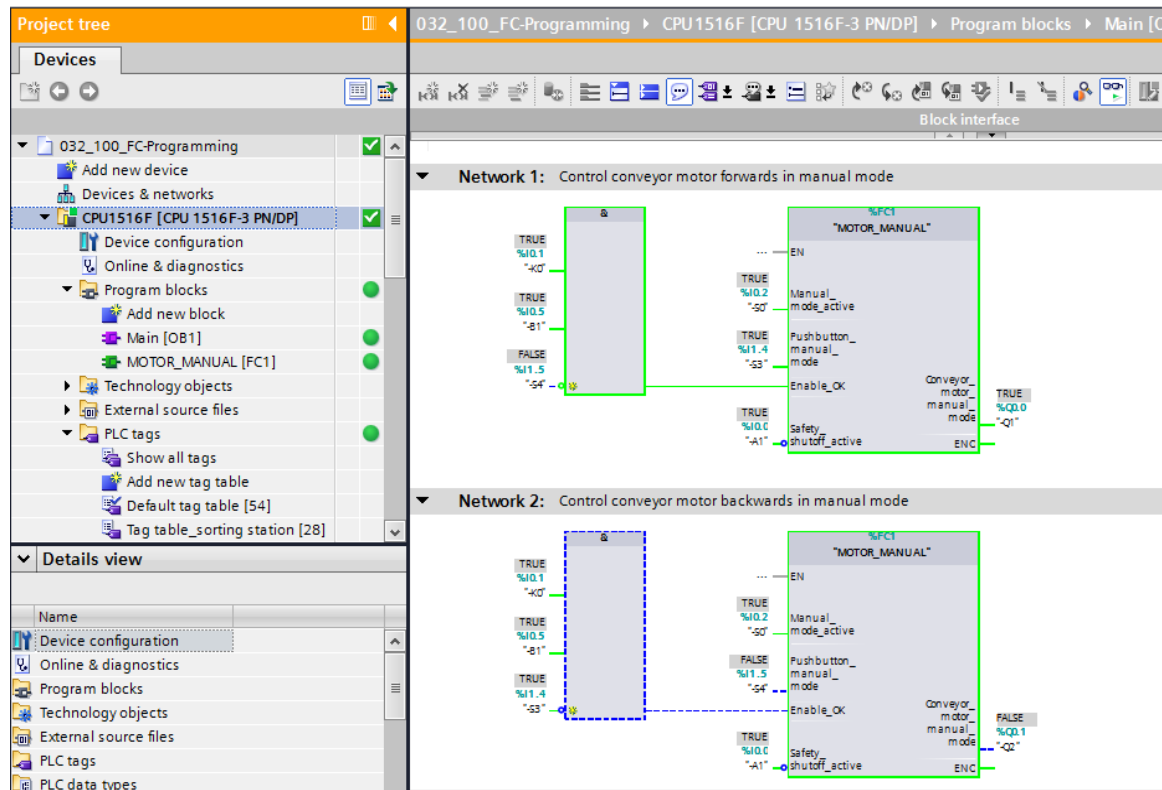


Figure 2: Online view of the Main [OB1] block

4 Task

The following diagnostic functions will be shown and tested in this chapter:

- Diagnostic symbols in the online view of the TIA Portal
- Device diagnostics with module status
- Offline/online comparison
- Monitoring and modifying tags
- Forcing tags

5 Planning

The diagnostic functions will be performed using a finished project as an example.

A project in the TIA Portal that was previously downloaded to the controller should be open for this.

In our case, after starting the TIA Portal, a previously created project will be retrieved from the archive and downloaded to the associated controller.

Afterwards, you can start implementing the diagnostic functions in the TIA Portal.

5.1 Online interface

Online diagnostics are only possible when the correct communication connection to the CPU has been set up beforehand. Here, we are connected via Ethernet/PROFINET.

When connecting online, you must therefore set the appropriate interfaces for your automation system.

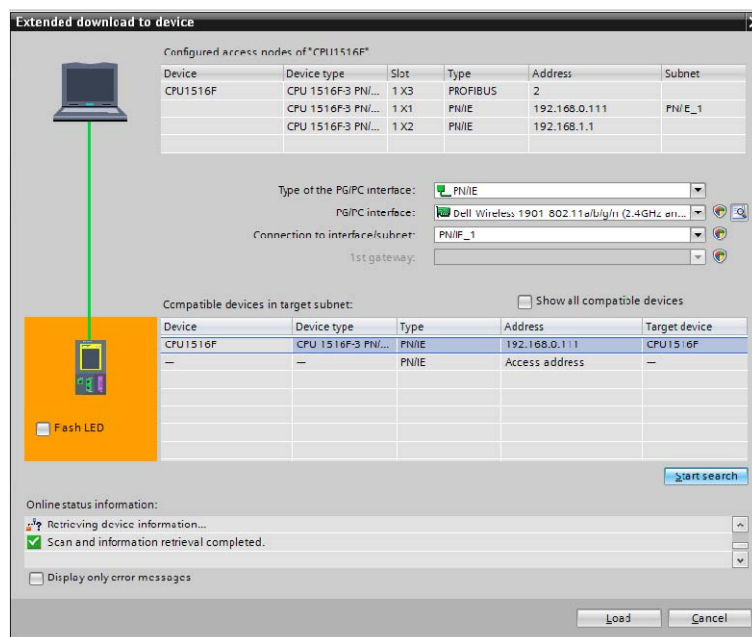


Figure 3: Go online

6 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

6.1 Retrieve an existing project

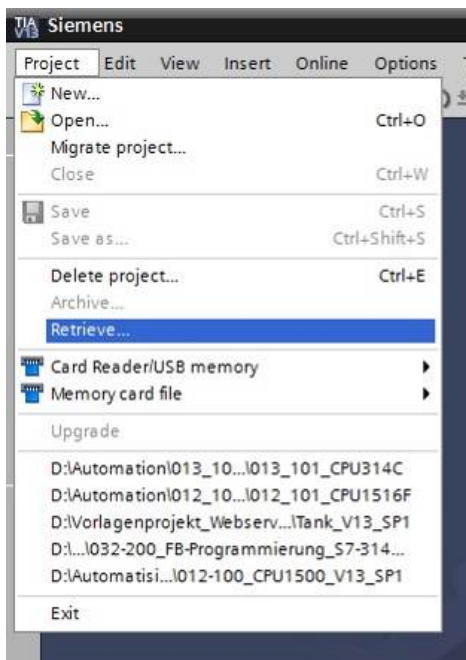
→ Before we can start the diagnostic functions, we need a project with programming and a hardware configuration.

(e.g., SCE_EN_032-100_FC-Programming....zap).

To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view.

Confirm your selection with "Open".


(→ Project → Retrieve → Select a .zap archive → Open)

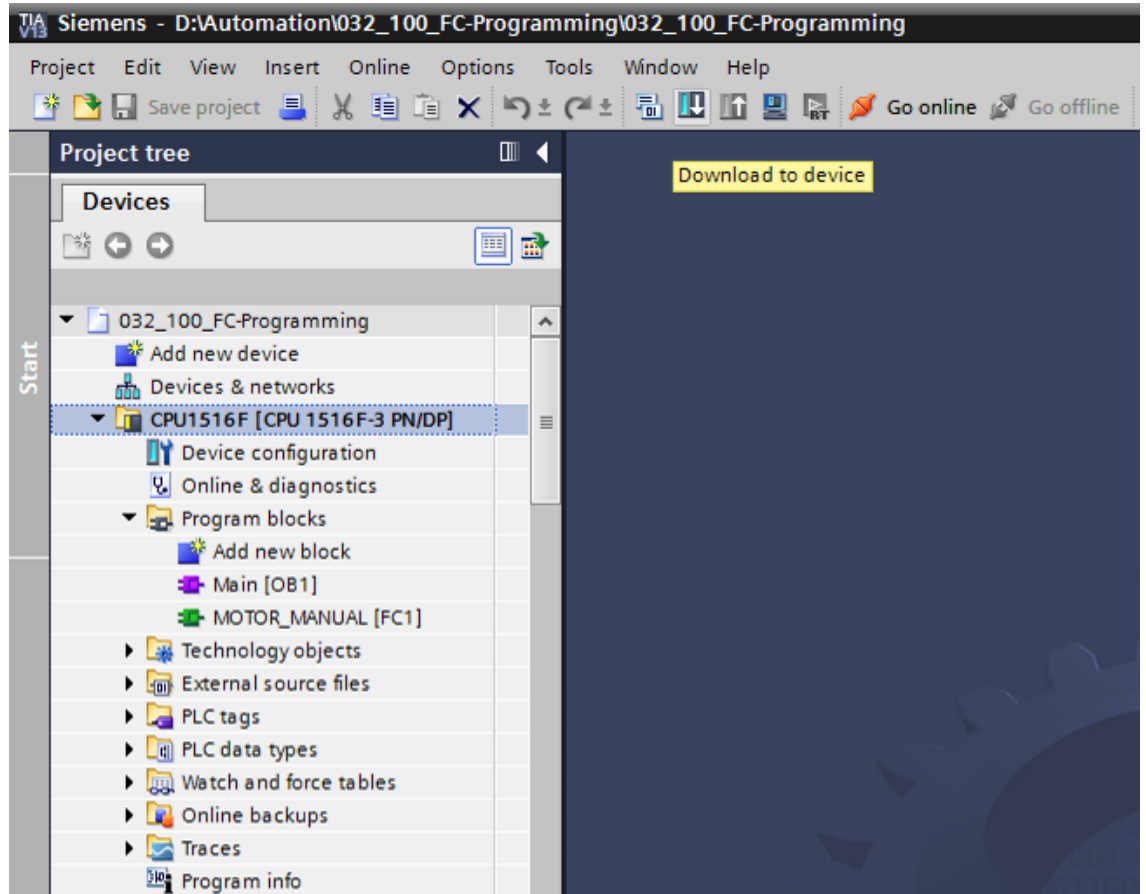


→ The next step is to select the target directory where the retrieved project will be stored.

Confirm your selection with "OK". (→ Target directory → OK)

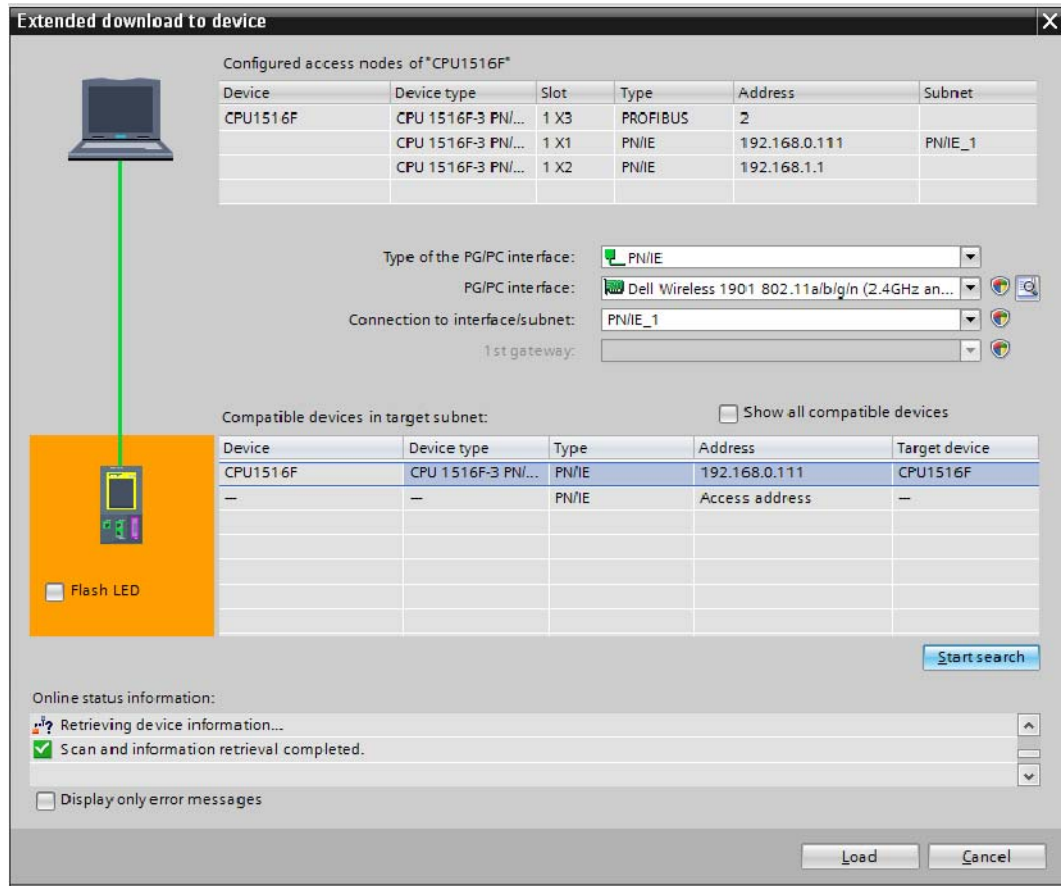
6.2 Download the program

→ After the project has been successfully retrieved, the controller can be selected and downloaded together with the created program. (→ )

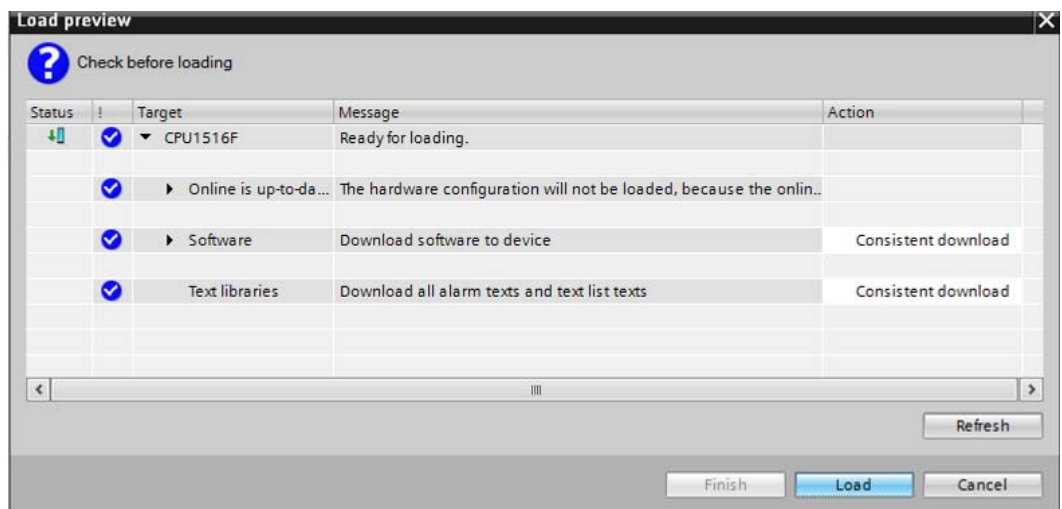


→ Select the correct interfaces and click "Start search". (→ "PN/IE" → Selection of the network adapter of the PG/PC → Direct at slot '1 X1' → "Start search")

Once "Scan and information retrieval completed" appears, click "Load". (→ "Load")

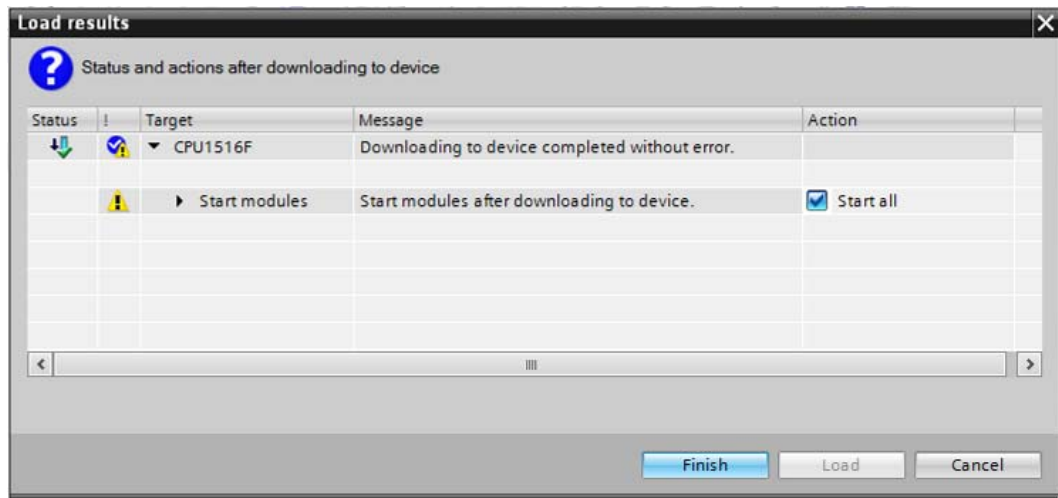


→ Before downloading can be started, other actions may have to be set (highlighted in pink). Click "Load" again. (→ "Load").



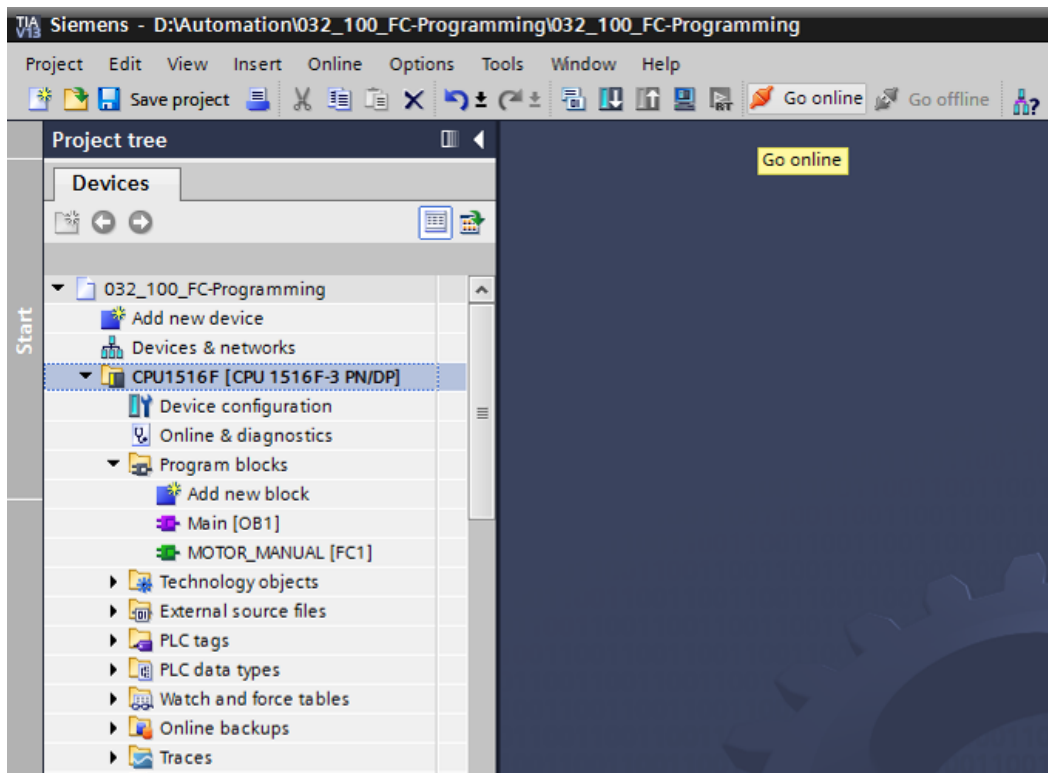
→ After loading, first select the "Start all" check box under Action.


Click "Finish". (→ select check box → "Finish")

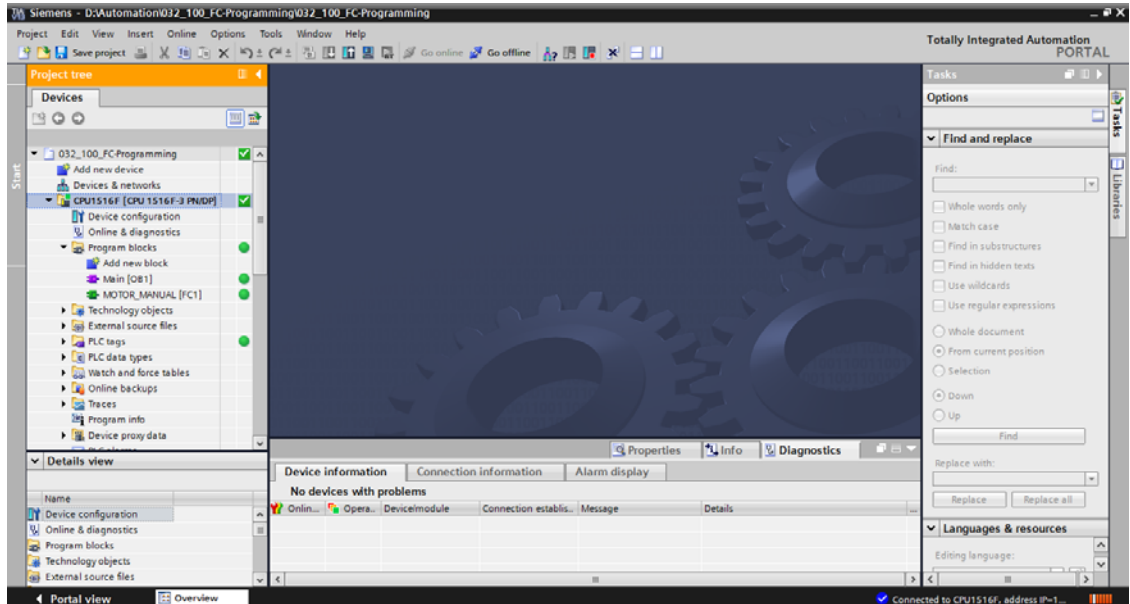


6.3 Go online

→ To get started with the diagnostic functions, we will select our controller ("PLC_1") and click "Go online". (→ PLC_1 → Go online)








- Once the online connection to the "PLC_1" controller is established, the CPU can be started or stopped with the following buttons . Diagnostic information in the form of symbols will already be available in the project tree and in the diagnostics window.



9

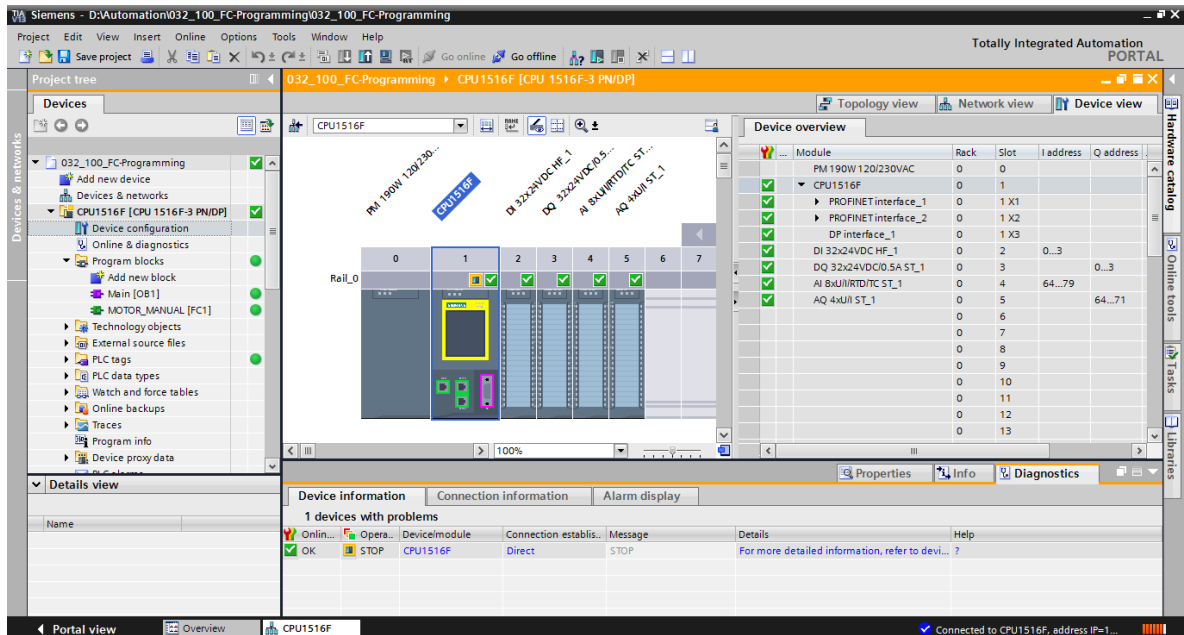
Symbols for the comparison status in the project tree

- The diagnostic symbols in the project tree show a comparison status representing the online/offline comparison of the project structure.

Symbol	Meaning
	Folder contains objects whose online and offline versions are different (only in the project tree)
	Online and offline versions of the object are different
	Object only exists online
	Object only exists offline
	Online and offline versions of the object are the same

→ Double-click "Device configuration".

(→ Device configuration)



















Operating state symbols for CPUs and CPs

→ The graphical representation and device information window show the various operating states of the CPU or communication processors (CPs).

Symbol	Operating state
	RUN
	STOP
	STARTUP
	HOLD
	DEFECT
	Unknown operating state
	The configured module does not support display of the operating state.




Diagnostic symbols for modules and devices in the device overview

- The graphical representation and Device overview window display the operating states of the various modules, CPU, or communication processors (CPs) using the following symbols.

Symbol	Meaning
	The connection to a CPU is currently being established.
	The CPU is not accessible at the set address.
	The type of CPU configured and type of CPU actually present are incompatible.
	During the establishment of the online connection to a protected CPU, the password dialog was terminated without input of the correct password.
	No fault
	Maintenance required
	Maintenance demanded
	Fault
	The module or device is deactivated.
	The module or device cannot be accessed from the CPU (valid for modules and devices below a CPU).
	Diagnostic data is not available because the current online configuration data differs from the offline configuration data.
	The configured module or device and the module or device actually present are incompatible (valid for modules or devices below a CPU).
	The configured module does not support display of the diagnostic status (valid for modules below a CPU).
	The connection has been established, but the state of the module is currently still being determined.
	The configured module does not support display of the diagnostic status.
	Error in lower-level component: A error is present in at least one lower-level hardware component.

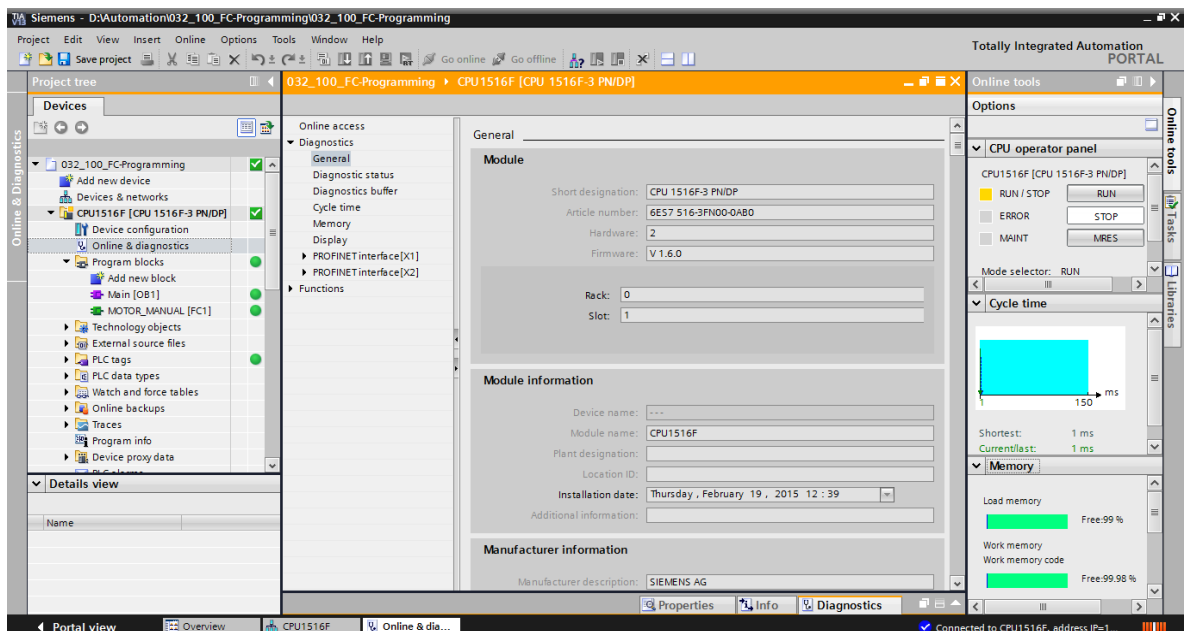
Color coding of ports and Ethernet cables

- The status of ports and Ethernet cables can be diagnosed in the Network view and Topology view.
- The following table shows the possible colors and their respective meaning.

Color	Meaning
	No fault or maintenance required
	Maintenance demanded
	Communication error

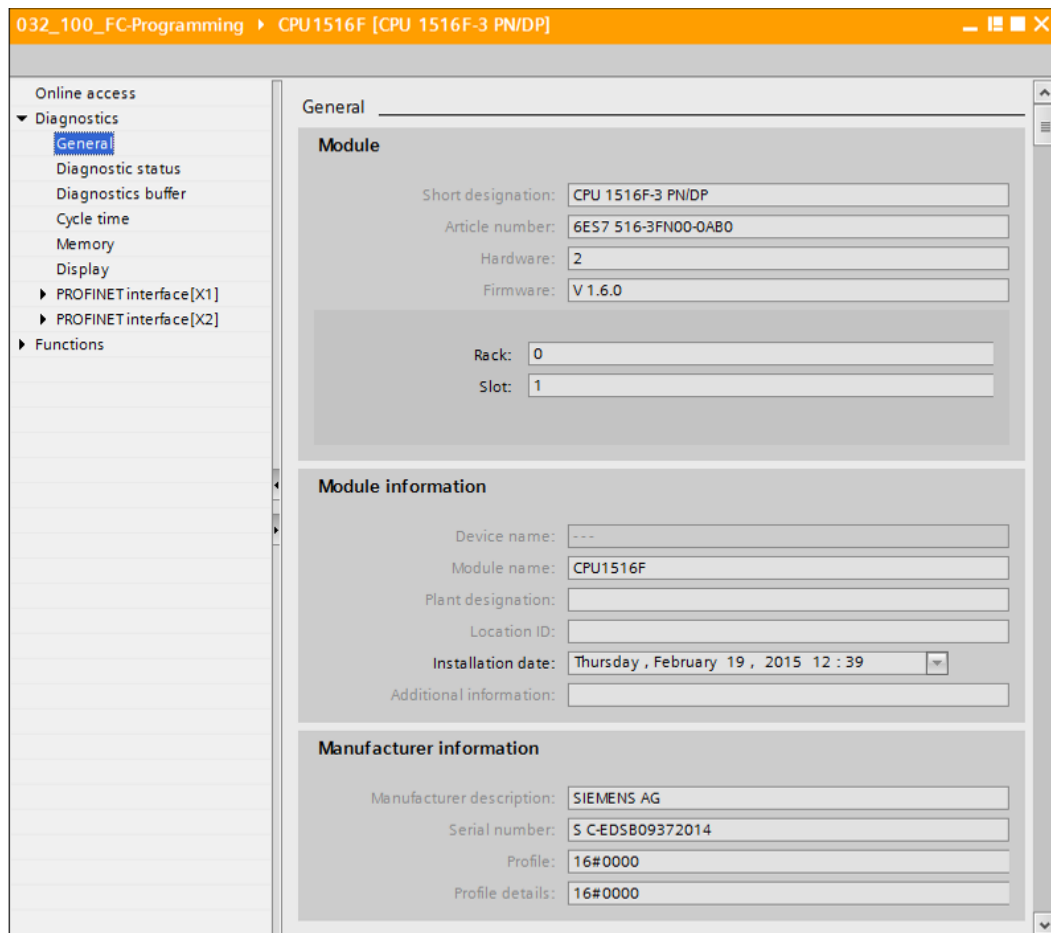
6.4 Online & diagnostics of the SIMATIC S7 controller

- Double-click "Online & diagnostics" in the project tree.
- (→ Online & diagnostics)
- A CPU operator panel, the cycle time and the memory utilization are displayed in the Online tools on the right side. Switch the CPU to RUN here. (→ RUN)

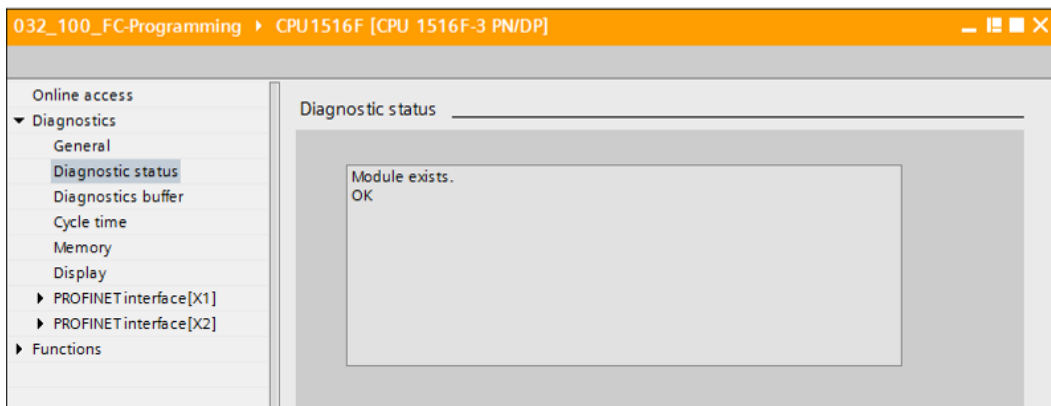


The screenshot shows the Siemens TIA Portal interface for online diagnostics. The main window displays the 'CPU1516F [CPU 1516F-3 PN/DP]' module information, including its designation, article number, hardware, and firmware. The 'Online tools' panel on the right shows the 'CPU operator panel' with buttons for RUN/STOP, ERROR, and MAINT. Below this, the 'Cycle time' and 'Memory' sections are visible, showing a graph of cycle time and memory usage bars.

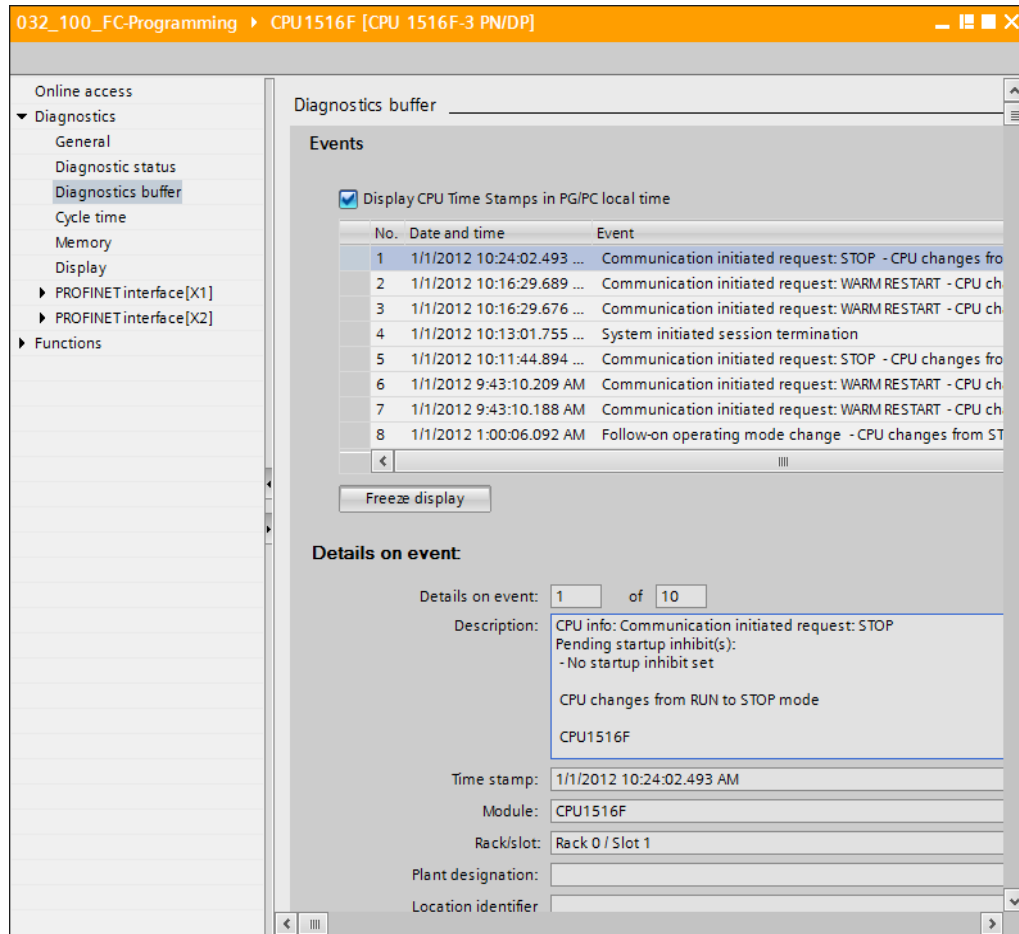
→ The working area window contains general information about the CPU. (→ General)



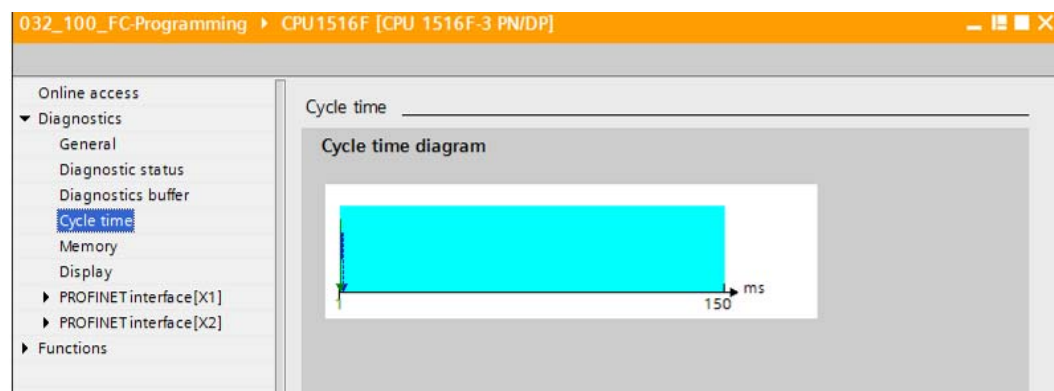
→ If diagnostic information is available, it is displayed in Diagnostic status.
(→ Diagnostic status)



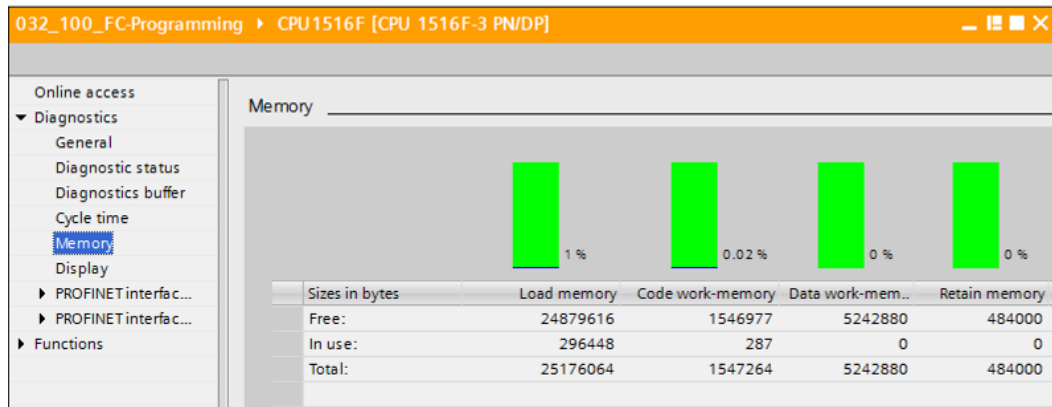
- Detailed Information on the individual events is displayed in Diagnostics buffer.
(→ Diagnostics buffer)



- Next you receive information about the cycle time of the executed program.
(→ Cycle time)



→ The memory utilization can be seen here in detail. (→ Memory)



→ Information about the display is also available for the CPU 1516F. (→ Display)

Module

Article number: 6ES7 591-1BA00-0AA0

Hardware: 98

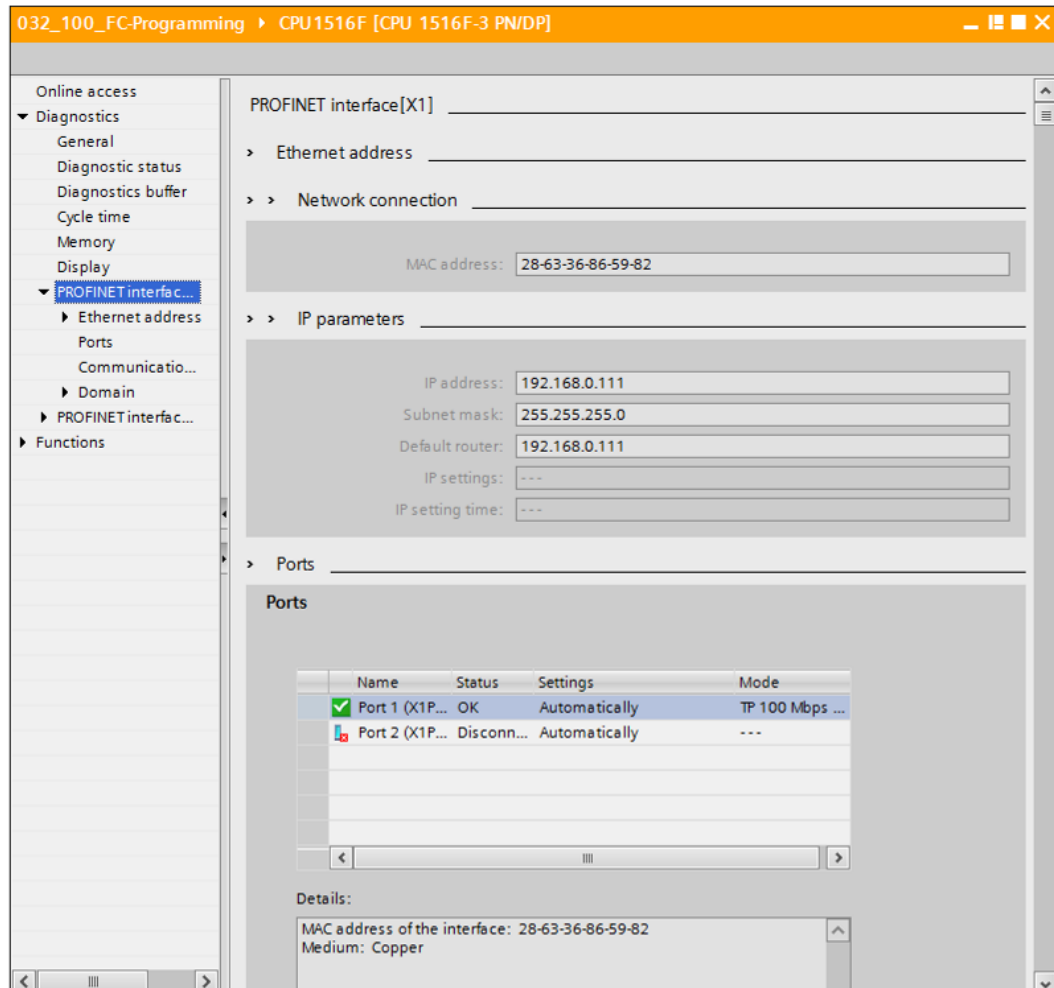
Firmware: V 1.6.0

Manufacturer description: SIEMENS AG

Serial number: S C-ENSJ6715

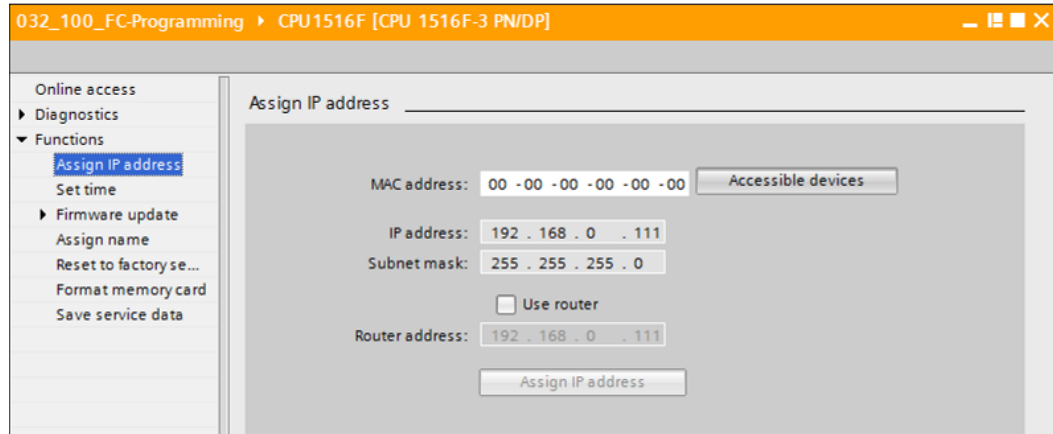
→ The network settings and the status of the PROFINET interfaces [X1] and [X2] can also be displayed.

(→ PROFINET interface [X1] or → PROFINET interface [X2])



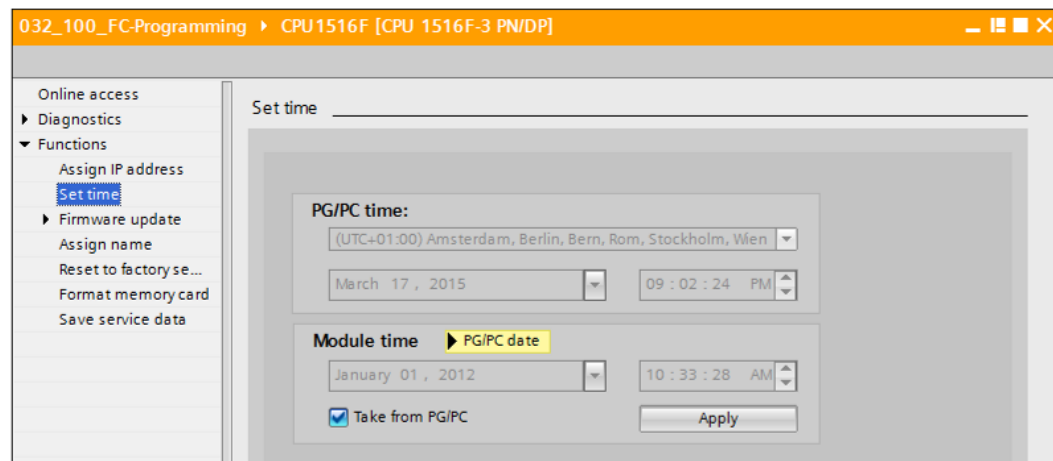
- In "Assign IP address" under Functions, you can assign the IP address to a controller. However, this is only possible when no hardware has been downloaded to the CPU.

(→ Functions → Assign IP address)



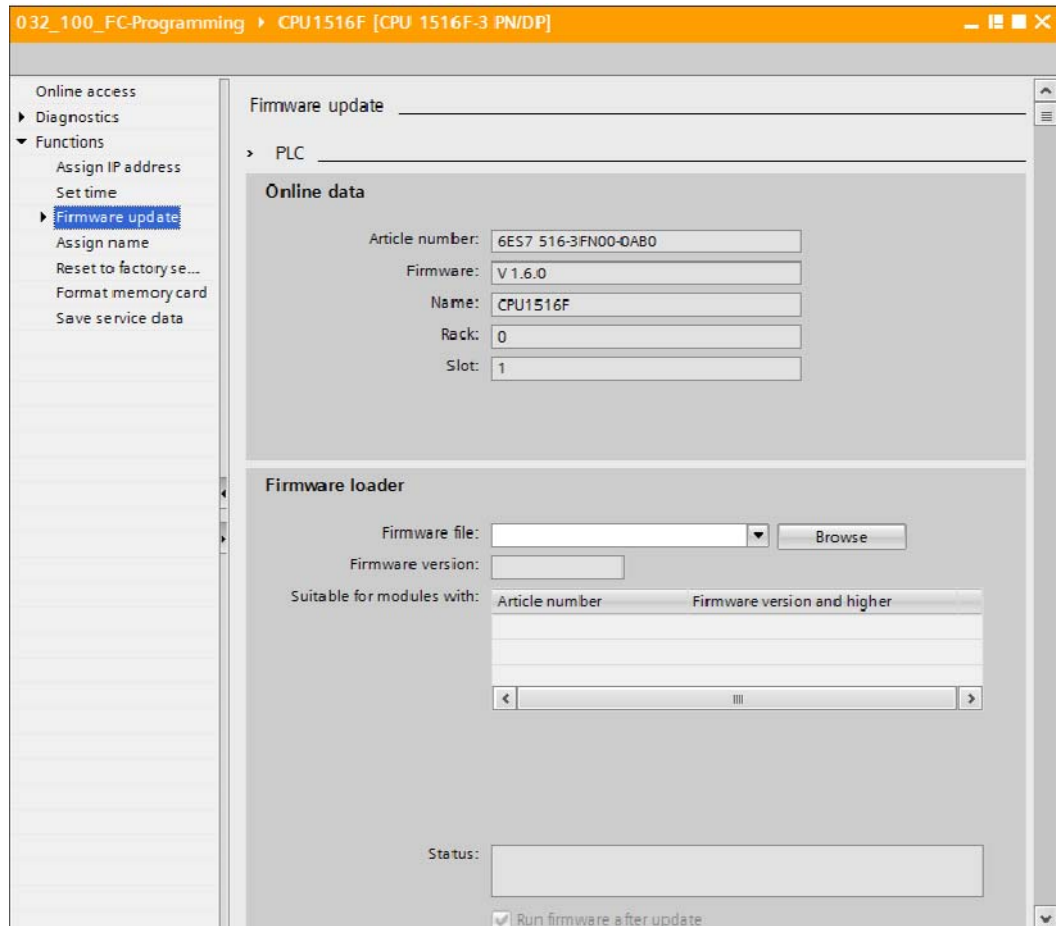
- Under "Set time", you can set the time of the CPU.

(→ Functions → Set time)



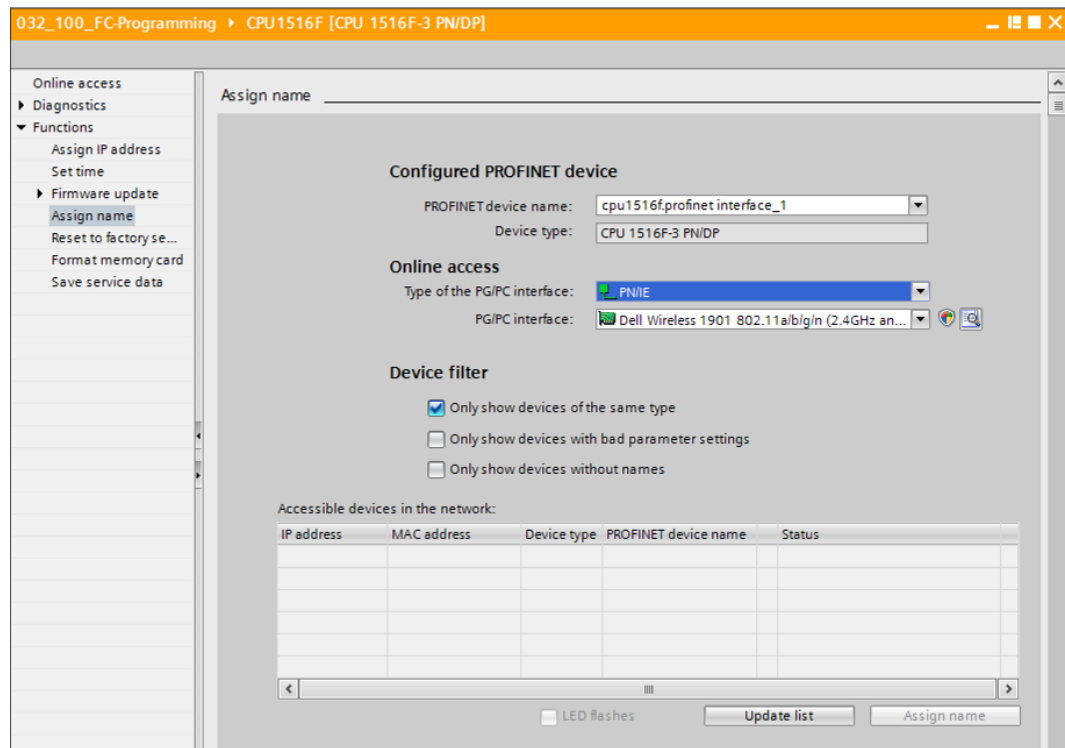
→ Under "Firmware update", you can update the firmware of the PLC or the display.

(→ Functions → Firmware update)



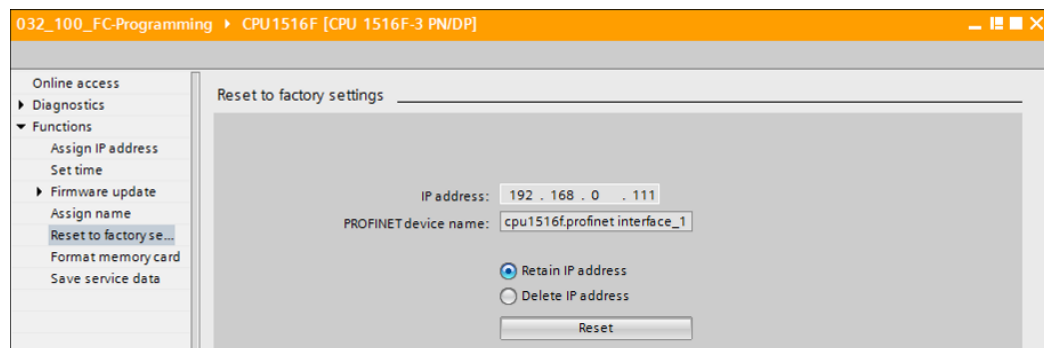
- Under "Assign name", you can assign a PROFINET device name to the configured field devices on PROFINET. The device name of the CPU cannot be changed here. It can only be changed by downloading a modified hardware configuration.

(→ Functions → Assign name)



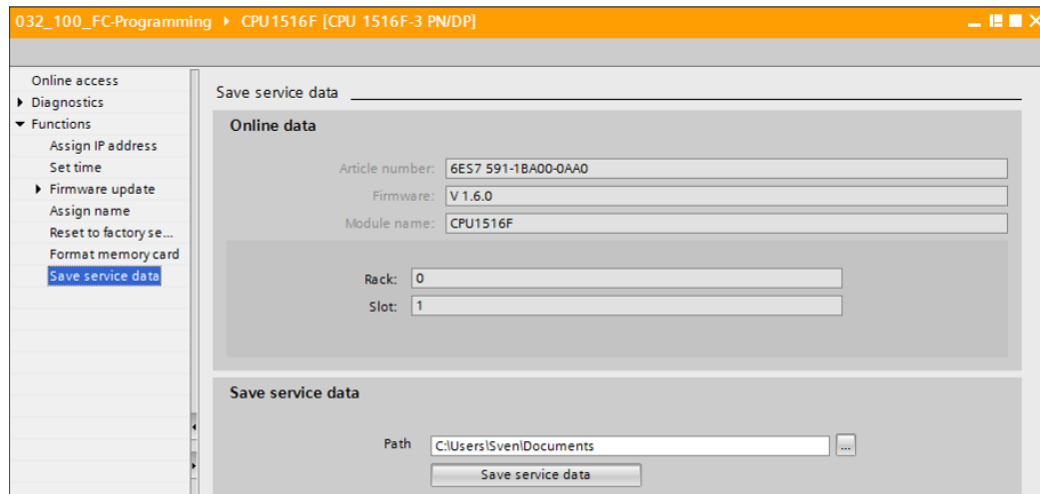
- Under "Reset to factory settings", you can restore the factory settings for the CPU. After restoring the factory settings, the CPU configuration and the program must be imported again from the inserted memory card. Therefore, the memory card must be formatted before the restoring the factory settings.

(→ Format memory card → Format → Reset to factory settings → Retain or delete IP address → Reset)



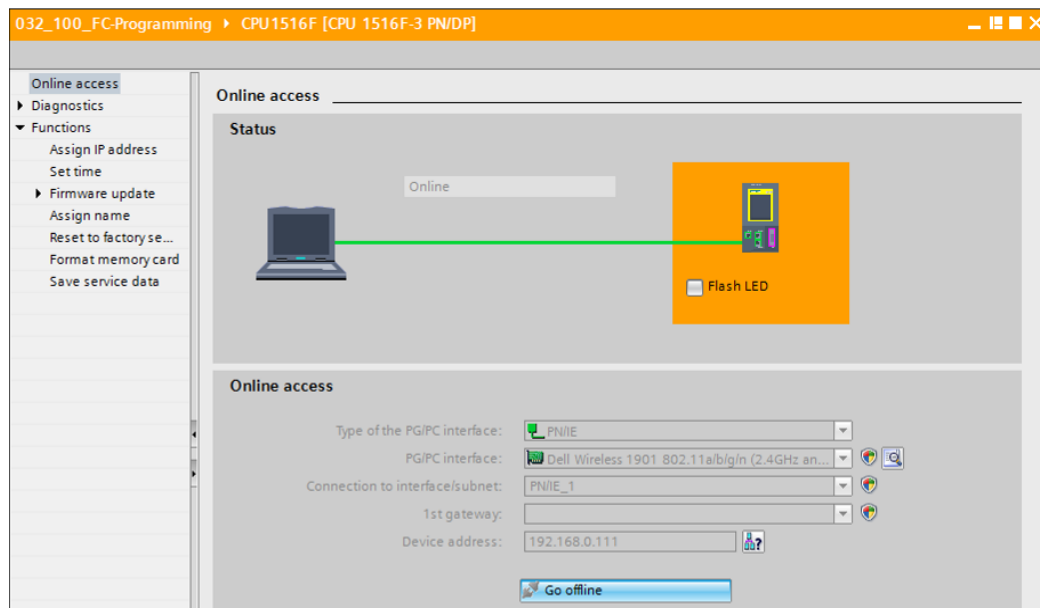
→ Finally, the service data can be saved under Functions.

(→ Functions → Save service data)



→ The online connection should be disconnected again before the next chapter.

(→ Online access → Disconnect online connection)



→ The TIA Portal is now back in offline mode. The orange-colored bars and the diagnostic symbols are no longer displayed.

6.5 Online/offline comparison

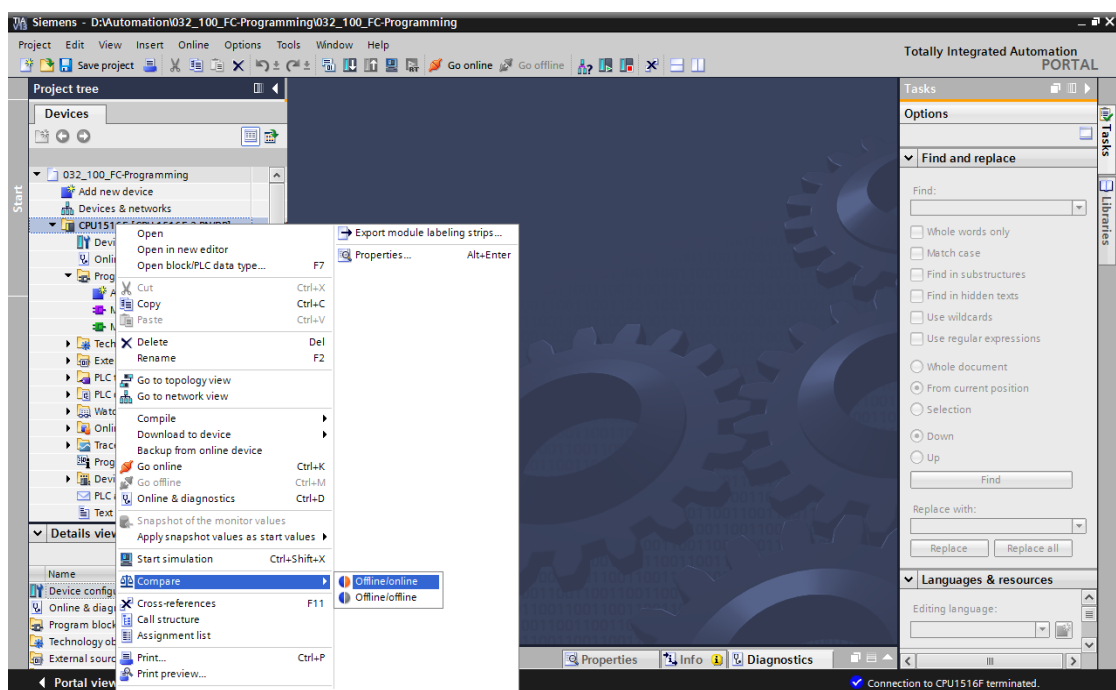
- It is often important to know whether the saved data matches the data loaded in the controller. First, remove the negation from the "Safety_shutoff_active" tag at the AND function in the "MOTOR_MANUAL [FC1]" block.

Save the "MOTOR_MANUAL [FC1]" block, but do **NOT** download it to the controller.

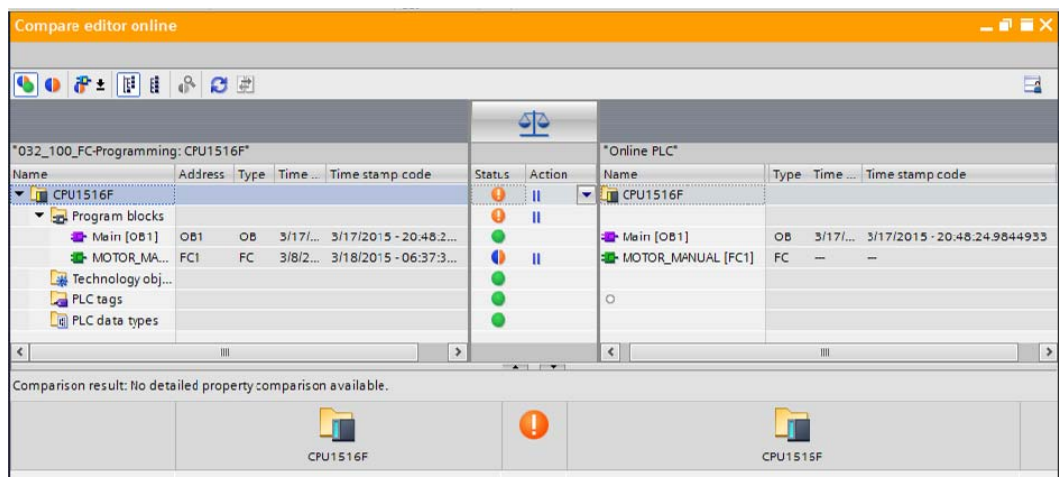
Close the "MOTOR_MANUAL [FC1]" block again.



- To compare, right-click the "PLC_1" controller and select "Compare", "Offline/online".

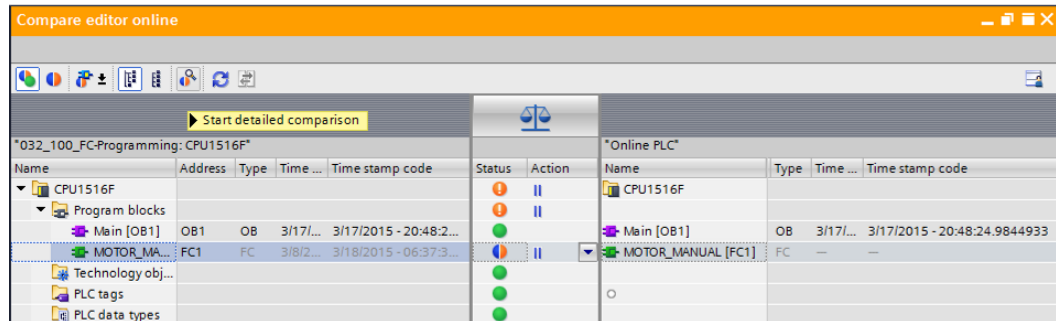
(→ Select controller → Compare → Offline/online)



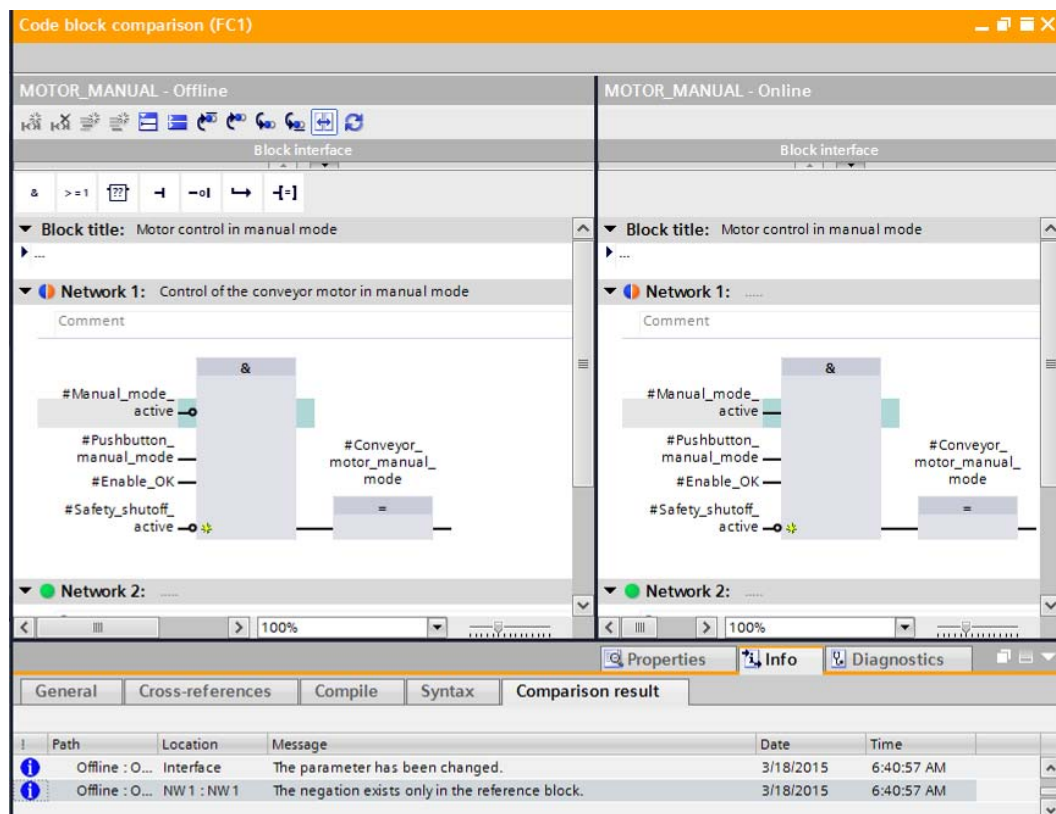
- The Compare editor online opens.



- If, for example, block differences are indicated , first select the block involved. You can then click the  button to "Start detailed comparison".
(→ MOTOR_MANUAL → Start detailed comparison).



- The selected offline/online block will be compared in the code block comparison. A detailed description of the difference is shown in the comparison result.

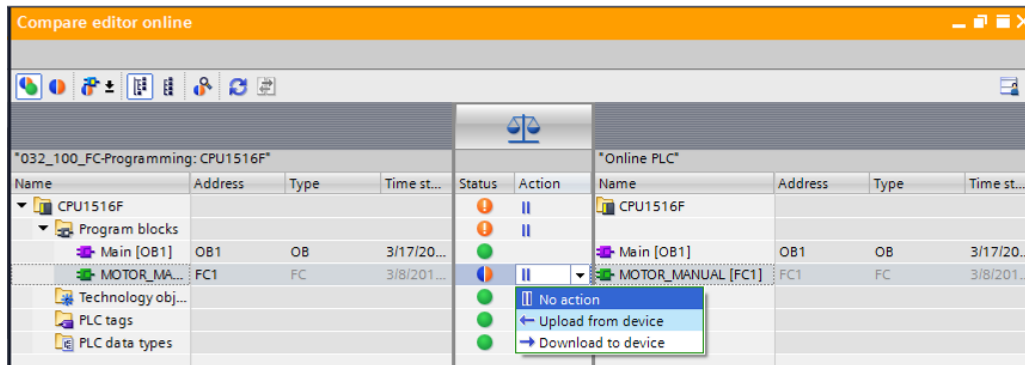



- Close the window of the code block comparison.

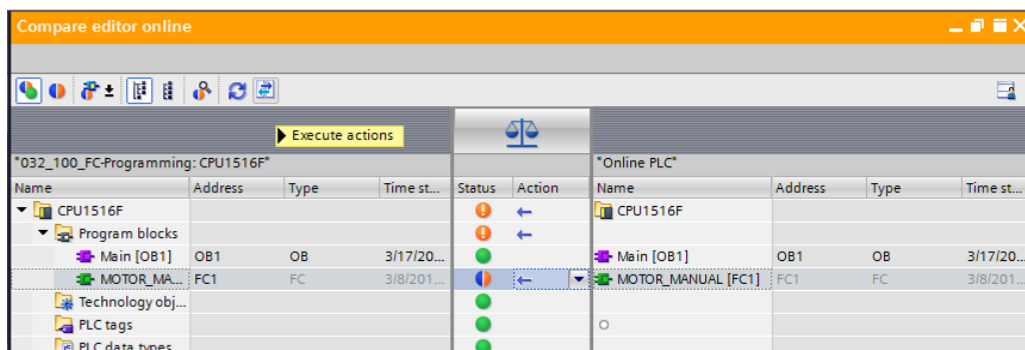
→ An action can be selected for the block involved in the Compare editor.

Either the "MOTOR_MANUAL" block will be downloaded from the programming device to the controller and overwritten there or the "MOTOR_MANUAL" block will be imported from the controller and overwritten in the TIA Portal.

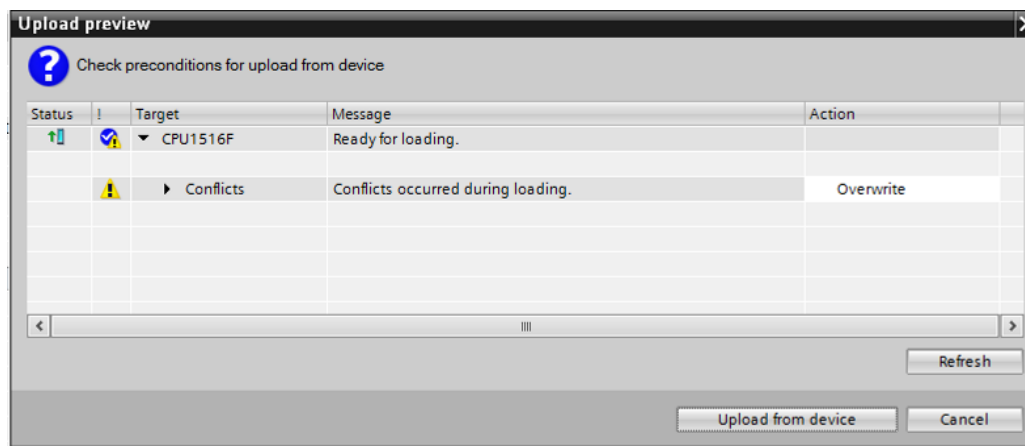
Select the "Upload from device" action. (← Upload from device)



→ Click the "Execute actions" button . (→ Execute actions)



→ Confirm "Upload from device". (→ Upload from device)



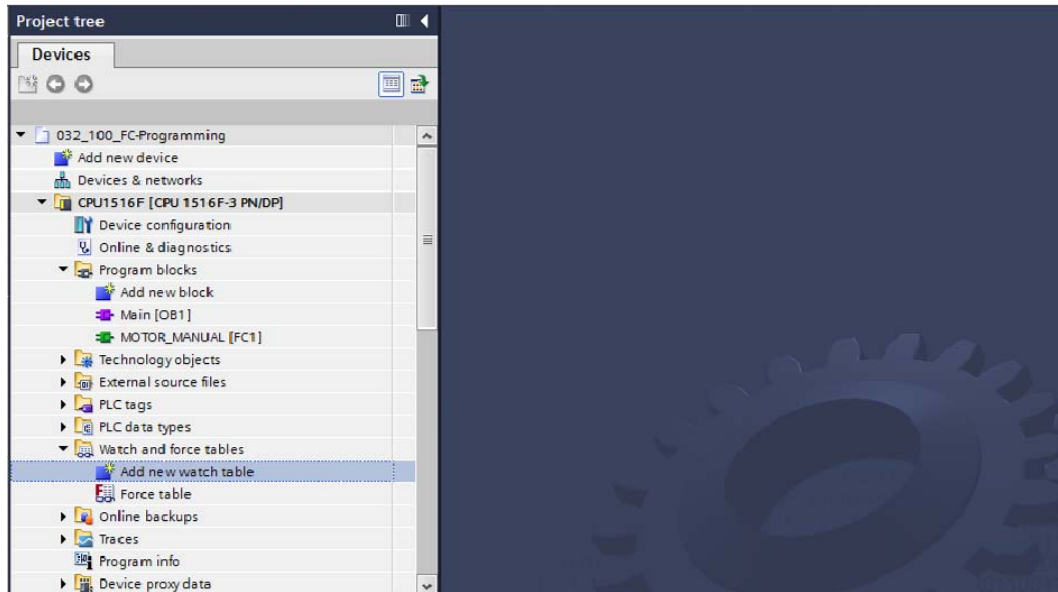
→ After the upload, there are no more differences. You should now save your project again and close the online connection.

6.6 Monitor and modify tags

→ To monitor and modify tags, you need a watch table.

Double-click "Add new watch table" in the project tree.

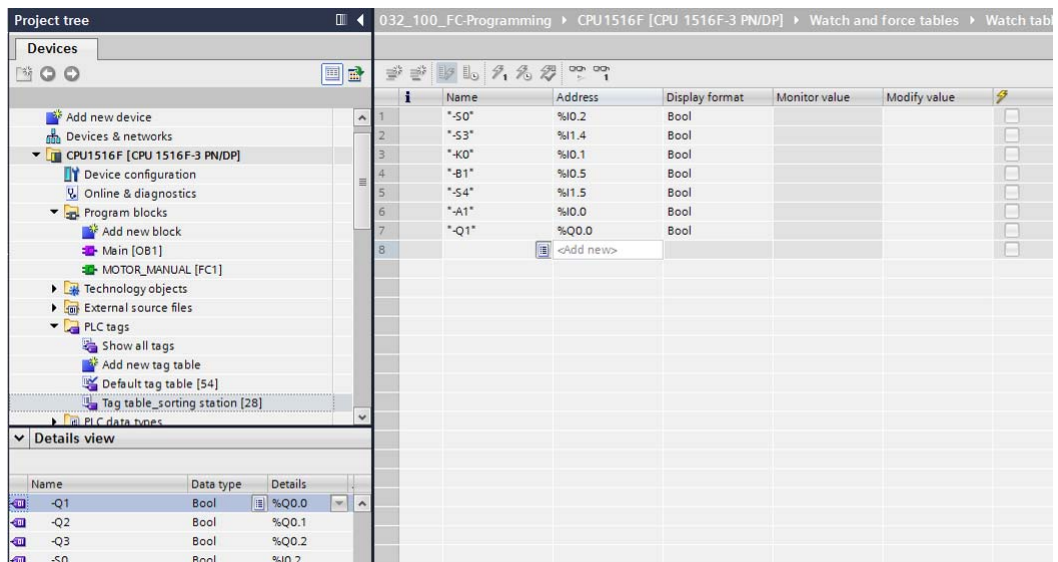
(→ Add new watch table)





→ Open the newly created "Watch_table_1" by double-clicking it. (→ "Watch_table_1")

→ You can enter individual tags in the table or you can select the "Tag_table_sorting_station" and the tags to be monitored, and drag them from the Details view to the watch table.

(→ Default tag table)

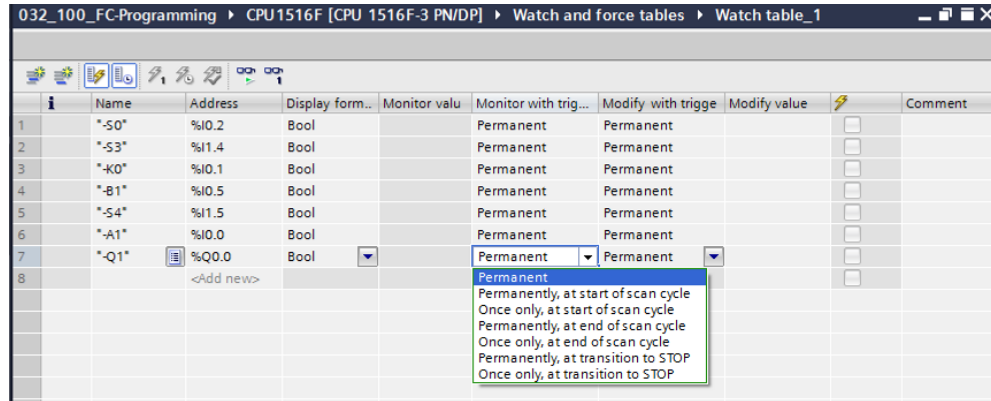


→ To have all monitoring and modifying functions available for selection, the following columns can be displayed:

'All modify columns'  and 'All expanded mode columns' .

Continue by selecting the trigger timing for the monitoring.

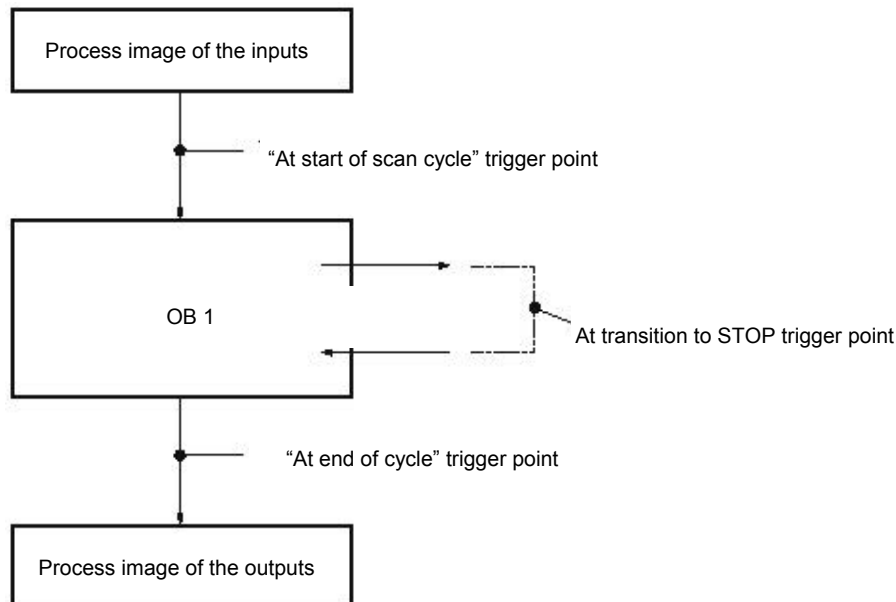
(→ Permanent)



	Name	Address	Display form...	Monitor val...	Monitor with trig...	Modify with trigge	Modify value		Comment
1	*-S0*	%I0.2	Bool		Permanent	Permanent		<input type="checkbox"/>	
2	*-S3*	%I1.4	Bool		Permanent	Permanent		<input type="checkbox"/>	
3	*-K0*	%I0.1	Bool		Permanent	Permanent		<input type="checkbox"/>	
4	*-B1*	%I0.5	Bool		Permanent	Permanent		<input type="checkbox"/>	
5	*-S4*	%I1.5	Bool		Permanent	Permanent		<input type="checkbox"/>	
6	*-A1*	%I0.0	Bool		Permanent	Permanent		<input type="checkbox"/>	
7	*-Q1*	%Q0.0	Bool		Permanent	Permanent		<input type="checkbox"/>	
8		<Add new>						<input type="checkbox"/>	



The following monitoring and modifying modes are available:


- Permanent (in this mode, the inputs are monitored/modified at the start of the cycle and the outputs at the end.)
- Once only, at start of scan cycle
- Once only, at end of scan cycle
- Permanently, at start of scan cycle
- Permanently, at end of scan cycle
- Once only, at transition to STOP
- Permanently, at transition to STOP



→ Next, click "Monitor all values once and now"  or "Monitor all values according to trigger settings" . (→  Monitor all)

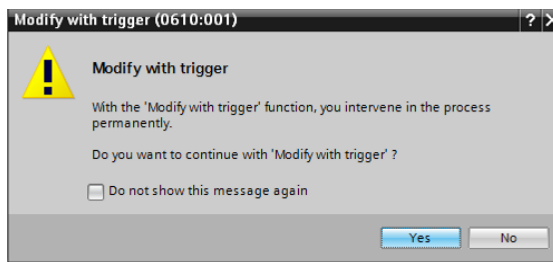
	Name	Address	Display form...	Monitor valu	Monitor with trig...	Modify with trigge	Modify value		Comment
1	*S0*	%I0.2	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
2	*S3*	%I1.4	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
3	*K0*	%I0.1	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
4	*B1*	%I0.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
5	*S4*	%I1.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
6	*A1*	%I0.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
7	*Q1*	%Q0.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
8	<Add new>								

→ To modify tags, enter the desired "Modify values". Next, click  to "Modify all activated values once and now" or  to "Modify all activated values by Modify with trigger condition".

(→ TRUE →  modifies all activated values by "Modify with trigger condition")

	Name	Address	Display form...	Monitor valu	Monitor with trig...	Modify with trigge	Modify value		Comment
1	*S0*	%I0.2	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
2	*S3*	%I1.4	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
3	*K0*	%I0.1	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
4	*B1*	%I0.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
5	*S4*	%I1.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
6	*A1*	%I0.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
7	*Q1*	%Q0.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	TRUE	<input checked="" type="checkbox"/>	
8	<Add new>								

→ Confirm the warning with 'Yes'. (→ Yes)



→ The output becomes active even though the programmed conditions are not met.

	Name	Address	Display form...	Monitor valu	Monitor with trig...	Modify with trigge	Modify value		Comment
1	*S0*	%I0.2	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
2	*S3*	%I1.4	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
3	*K0*	%I0.1	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
4	*B1*	%I0.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
5	*S4*	%I1.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
6	*A1*	%I0.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent		<input type="checkbox"/>	
7	*Q1*	%Q0.0	Bool	<input checked="" type="checkbox"/> TRUE	Permanent	Permanent	TRUE	<input checked="" type="checkbox"/>	
8	<Add new>								

Note: If the watch table is closed or the connection to the PLC is lost, all modify commands become ineffective.

6.7 Force tags

- The "Force" function can be used to assign a fixed value to tags. Force values are specified in a similar way as for the "Modify tags" function but, in contrast, are retained after the CPU is stopped. The main differences between "Modify tags" and the "Force" function are as follows: In contrast with "Modify tags", the "Force" function does not allow you to assign values to data blocks, timers, counters and bit memory.

I/O device inputs (e.g., IWxx:P) cannot be modified, although they can be pre-assigned by the "Force" function.

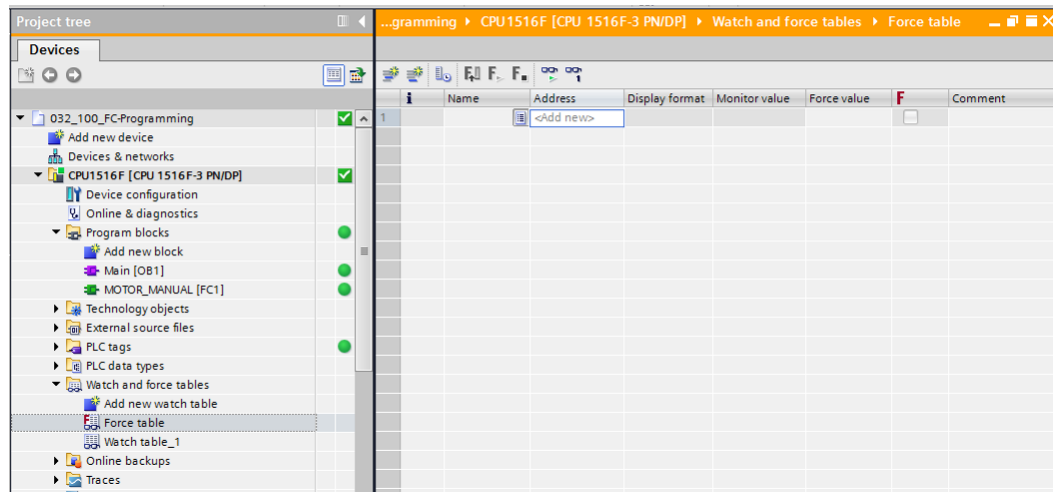
Unlike with the "Modify" function, values permanently assigned by the "Force" function cannot be overwritten by the user program.

If you close the force table, the force values are retained. This is not the case with the "Modify" function.

If the online connection to the CPU is interrupted, the tags assigned with the "Force" function retain their value.

- To force tags, you must first double-click the force table to open it.

(→ Force table)



→ Select the "Q1" operand with address %Q0.0 from the list. (→ Q1)

	Name	Address	Display format	Monitor value	Force value	F	Comment
1	<Add new>						
	*-P7"		Bool	%Q1.3	display cylinder...		
	*-Q1"		Bool	%Q0.0	conveyor motor...		
	*-Q2"		Bool	%Q0.1	conveyor motor...		
	*-Q3"		Bool	%Q0.2	conveyor motor...		
	*-S0"		Bool	%I0.2	mode selector ...		
	*-S1"		Bool	%I0.3	pushbutton aut...		
	*-S2"		Bool	%I0.4	pushbutton aut...		
	*-S3"		Bool	%I1.4	pushbutton ma...		

→ With forcing, the operands are entered with direct I/O access (%Q0.0:P).

	Name	Address	Display format	Monitor value	Force value	F	Comment
1	*-Q1":P	%Q0.0:P	Bool				
2	<Add new>						

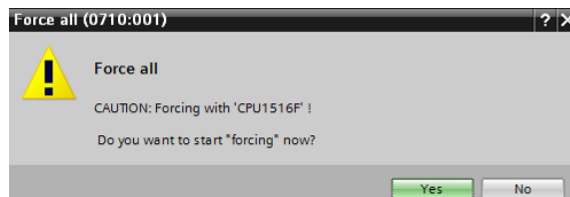
→ Enter the desired force value and activate it

Click "Start or replace forcing" . The new force request is transferred to the CPU.

(→ %Q0.0:P → TRUE → → Start or replace forcing)

	Name	Address	Display format	Monitor value	Force value	F	Comment
1	*-Q1":P	%Q0.0:P	Bool		TRUE		
2	<Add new>						


→ Confirm the warning with 'Yes'. (→ Yes)









→ Forcing is activated and the yellow **MAINT LED** on the CPU lights up. In addition, an **F** on a red background is shown at the top right of the display of the S7-1500.

	Name	Address	Display format	Monitor value	Force value	F	Comment
1	*-Q1":P	%Q0.0:P	Bool		TRUE		


Note: If the watch table is closed or the connection to the PLC is lost, **forcing remains active**, and the yellow **FRCE LED** on the CPU continues to be lit.










→ If you want to 'Stop forcing', simply click "Stop forcing" , and confirm the next dialog with "Yes".



(→  Stop forcing) 'Yes' (→ Yes)










	i	Name	Address	Display format	Monitor value	Force value		Comment
1		*-Q1*-P	%Q0.0:P	Bool		TRUE	 	
2			<Add new>					

Forcing is stopped and the yellow **MAINT LED** on the CPU goes out.

→ If there is already a force request in the controller, this is indicated by the  symbol in the watch table.

...mming > CPU1516F [CPU 1516F-3 PN/DP] > Watch and force tables > Watch table_1								
	i	Name	Address	Display form...	Monitor valu	Monitor with trig...	Modify with trigge	Modify v...
1		*-S0*	%I0.2	Bool	 FALSE	Permanent	Permanent	
2		*-S3*	%I1.4	Bool	 FALSE	Permanent	Permanent	
3		*-K0*	%I0.1	Bool	 FALSE	Permanent	Permanent	
4		*-B1*	%I0.5	Bool	 FALSE	Permanent	Permanent	
5		*-S4*	%I1.5	Bool	 FALSE	Permanent	Permanent	
6		*-A1*	%I0.0	Bool	 FALSE	Permanent	Permanent	
7		*-Q1*	%Q0.0	Bool	 FALSE	Permanent	Permanent	
8			<Add new>					

→ If you now click , additional information is displayed. (→ )

...mming > CPU1516F [CPU 1516F-3 PN/DP] > Watch and force tables > Watch table_1								
	i	Name	Address	Display form...	Monitor valu	Monitor with trig...	Modify with trigge	Modify v...
1		*-S0*	%I0.2	Bool	 FALSE	Permanent	Permanent	
2		*-S3*	%I1.4	Bool	 FALSE	Permanent	Permanent	
3		*-K0*	%I0.1	Bool	 FALSE	Permanent	Permanent	
4		*-B1*	%I0.5	Bool	 FALSE	Permanent	Permanent	
5		*-S4*	%I1.5	Bool	 FALSE	Permanent	Permanent	
6		*-A1*	%I0.0	Bool	 FALSE	Permanent	Permanent	
7		*-Q1*	%Q0.0	Bool	 FALSE	Permanent	Permanent	
8			<Add new>					

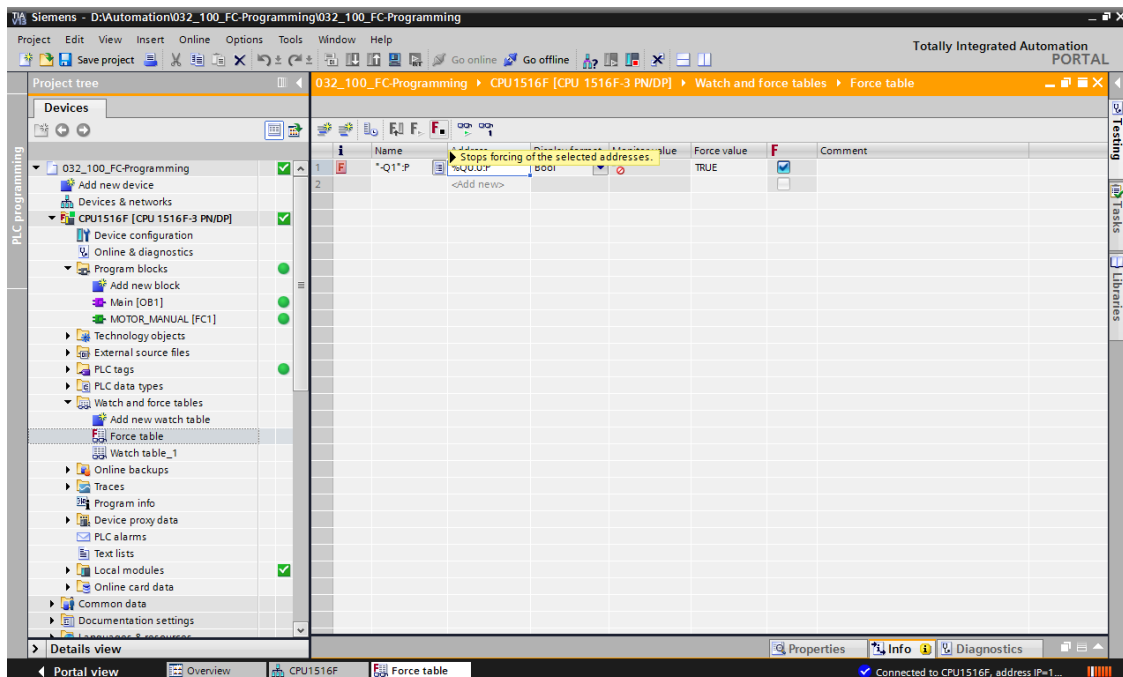
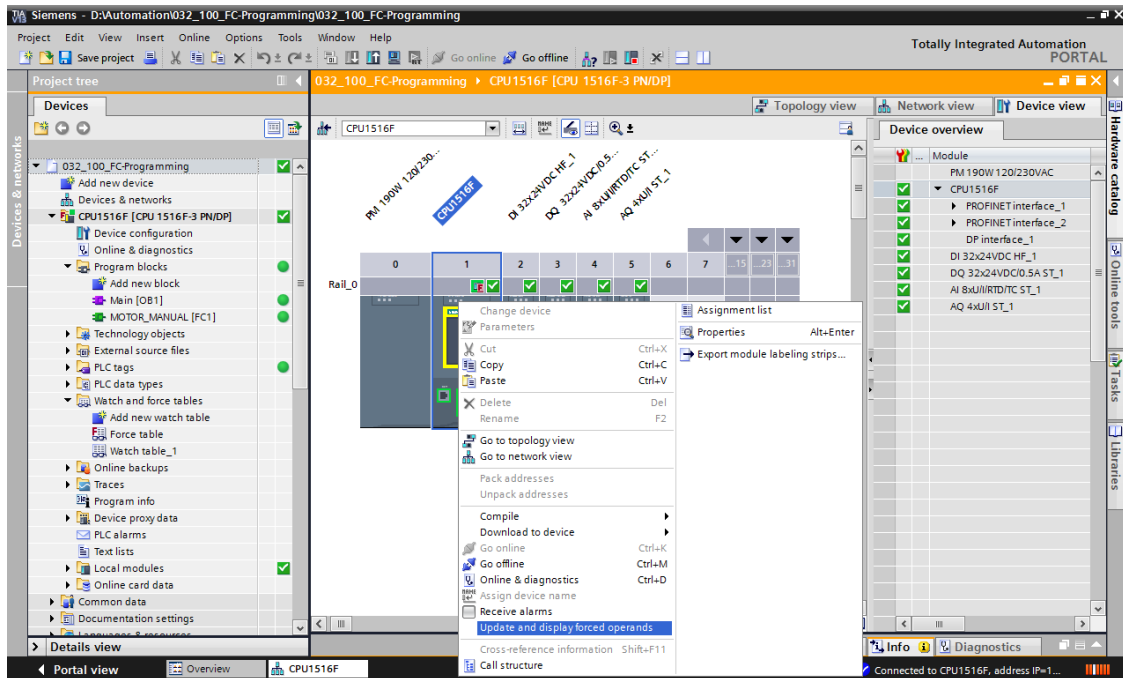
The I/O belonging to address '%Q0.0' will be forced with the value 'TRUE'.

→ If there is already a force request in the controller, it can also be displayed and stopped via the online device view. For this, you need to right-click the CPU in online mode of the device view and select "Update and display forced operands".

(→ right-click the CPU → Update and display forced operands")

→ The force table with the current force requests will now be displayed and you can stop these.

(→  Stop forcing)



6.8 Checklist

No.	Description	Completed
1	Project 032-100_FB-programming... successfully retrieved.	
2	CPU 1516F from project 032-100_FB-Programming... successfully downloaded.	
3	CPU 1516F connected online.	
4	Status of the CPU 1516F checked with Online & diagnostics.	
5	Offline/online comparison of blocks in the CPU 1516F performed.	
6	Watch_table_1 created.	
7	Tags (-S0 / -S3 / -K0 / -B1 / -S4 / -A1 / -Q1) entered in watch table.	
8	Switch on conveyor motor forwards by modifying the output (-Q1 = 1) in watch table.	
9	Switch off conveyor motor forwards by modifying the output (-Q1 = 0) in watch table.	
10	Open force table	
11	Tag (-Q1:P) entered in force table.	
12	Switch on conveyor motor forwards by forcing the output (-Q1 = 1) in force table.	
13	Force output -Q1 to switch off again.	

7 Exercise

7.1 Task – Exercise

In this exercise, the MOTOR_AUTO [FB1] function block from chapter SCE_EN_032-200_FB-Programming is to be tested.

The problem here is that the cylinder is in the front end position and thus the enable conditions for switching on the conveyor are not OK.

Using a watch table, the cylinder is to be moved to its rear end position so that the enable conditions for the MOTOR_AUTO [FB1] block become OK.

7.2 Planning

Plan the implementation of the task independently using the step-by-instructions as an aid.

7.3 Checklist – Exercise

No.	Description	Completed
1	Project 032-200_FB-Programming... successfully retrieved.	
2	CPU 1516F from project 032-200_FB-Programming... successfully downloaded.	
3	Watch table created and renamed as "Watch_table_cylinder".	
4	Tags (-B1 / -B2 / -M2) entered in watch table.	
5	Retract cylinder by modifying the output (-M2 = 1) in watch table.	
6	Cylinder retracted (-B1 = 1)	
7	Reset output for Retract cylinder in watch table again (-M2 = 0).	

8 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

[illegible]

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Please note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, please contact your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, especially Prof. Dr.-Ing. Leon Urbas und Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	4
2	Prerequisite.....	4
3	Theory.....	4
3.1	System diagnostics: Automated creation of error messages	4
3.2	Diagnostics via web server	5
3.3	Diagnostics with the integrated display	6
4	Task	7
5	Planning.....	7
6	Structured step-by-step instructions	8
6.1	Retrieve an existing project	8
6.2	Configure the web server	9
6.3	Configure the display	13
6.4	Configure system diagnostics.....	14
6.5	Activate the diagnostics of the power supply for the anal. output module/download the PLC... 15	
6.6	Trigger error message	17
6.7	Display alarms in Online & diagnostics	18
6.8	Diagnostics for the S7-1500 via the web	20
6.9	Diagnostics for the S7-1500 via the integrated display	29
6.10	Checklist	30
7	Additional information	31

WEB SERVER AND ADVANCED DIAGNOSTICS

1 Goal

In this module, the reader will become acquainted with additional tools that are helpful for troubleshooting.

In particular, we will show you how automated alarm texts can be generated in the TIA Portal for hardware faults and system errors. These can then be displayed not only in the TIA Portal but also on the display of the CPU as well as via the web server of the CPU 1516F-3 PN/DP. It is also possible to bring these into the message windows of HMI systems for viewing.

This module will present advanced diagnostic functions that, for example, you can test with the TIA project from the SCE_EN_032-410_Basics_Diagnostics with the SIMATIC S7-1500 module.

2 Prerequisite

This chapter builds on the hardware configuration of the SIMATIC S7 CPU1516F-3 PN/DP. However, other hardware configurations can be used. You can use the following project for this chapter, for example:

SCE_EN_032-410_Basics_Diagnostics_2_R1503.zap13

3 Theory

3.1 System diagnostics: Automated creation of error messages

In the TIA Portal, the diagnostics of devices and modules is collectively referred to as system diagnostics. The monitoring functions are automatically derived from the hardware configuration.

All SIMATIC products have integrated diagnostic functions which you can use to detect and remedy faults. The components automatically signal a possible disruption of operation and provide additional detailed information. Undesired downtimes can be minimized with plant-wide diagnostics.

The following states are monitored by the system in the running plant:

- Device failure
- Pull/plug error
- Module fault
- I/O access error
- Channel fault
- Parameter assignment error
- Failure of the external auxiliary voltage

3.2 Diagnostics via web server

The web server enables monitoring and administering of the CPU by authorized users over a network.

This permits evaluation and diagnostics over long distances. Monitoring and evaluation is possible without the TIA Portal; all you need is a web browser.

The web server is deactivated in the delivery state of the CPU. This means that you must load a project in which the web server is activated to enable access using the web browser.

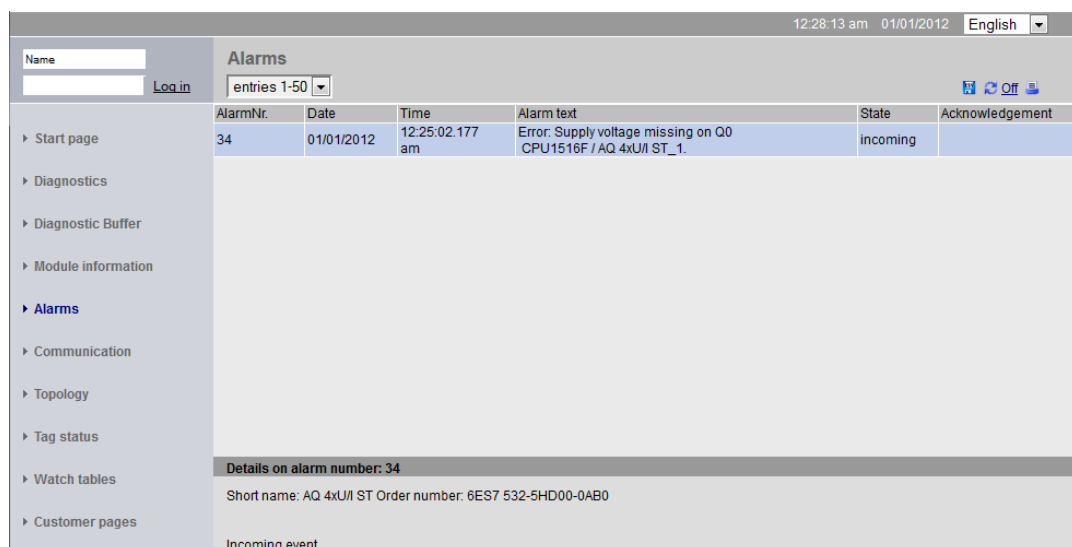
The web server offers the following security functions:

- Access via secure "https" transmission protocol
- User authorization by means of a user list
- Restriction of access from certain interfaces

You need a web browser to access the HTML pages of the CPU.

The following web browsers have been tested for communication with the CPU:

- Internet Explorer (Version 8)
- Mozilla Firefox (Version 21)
- Mobile Safari (iOS5)



The screenshot shows a web browser interface for a Siemens CPU. At the top, the status bar displays '12:28:13 am 01/01/2012' and 'English'. On the left is a navigation menu with options: Start page, Diagnostics, Diagnostic Buffer, Module information, **Alarms** (selected), Communication, Topology, Tag status, Watch tables, and Customer pages. The main content area is titled 'Alarms' and shows a table with one entry. Below the table, there is a section for 'Details on alarm number: 34'.

AlarmNr.	Date	Time	Alarm text	State	Acknowledgement
34	01/01/2012	12:25:02.177 am	Error: Supply voltage missing on Q0 CPU1516F / AQ 4xUI ST_1.	incoming	

Details on alarm number: 34
Short name: AQ 4xUI ST Order number: 6ES7 532-5HD00-0AB0
Incoming event

Figure 1: Web server of the CPU 1516F-3 PN/DP with alarm text from the system diagnostics

Note: Make sure that you protect the CPU from manipulation and unauthorized access through the use of different methods (e.g., limiting network access, using firewalls).

3.3 Diagnostics with the integrated display

The S7-1500 CPU has a front flap with a display and control keys. Control data and status data can be displayed in various menus on the display and numerous settings can be made. You use the control keys to navigate through the menus.

The display of the CPU offers the following functions:

- 6 different display languages can be selected.
- Diagnostic messages are displayed in plain text.
- The interface settings can be changed locally.
- Password assignment for display operation is possible through the TIA Portal.



Figure 2: Display of the CPU 1516F-3 PN/DP with alarm text from the system diagnostics

4 Task

The following advanced diagnostic functions will be shown and tested in this chapter:

- Configuration of web server of the CPU 1516F-3 PN/DP
- Configuration of display of the CPU 1516F-3 PN/DP
- Create hardware fault and system error alarms with the system diagnostics
- Display alarms via the web server of the CPU 1516F-3 PN/DP
- Display alarms via the integrated display of the CPU 1516F-3 PN/DP

5 Planning

The diagnostic functions will be performed using a finished project as an example. A project in the TIA Portal that was previously downloaded to the controller should be open for this. In our case, once you have started the TIA Portal, you will retrieve a previously created project that was archived and download it to the associated controller.

You can then configure the web server, the display and the system diagnostics in the TIA Portal. To test the system diagnostics, we will disconnect the monitored analog output module from its supply voltage.

6 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

6.1 Retrieve an existing project

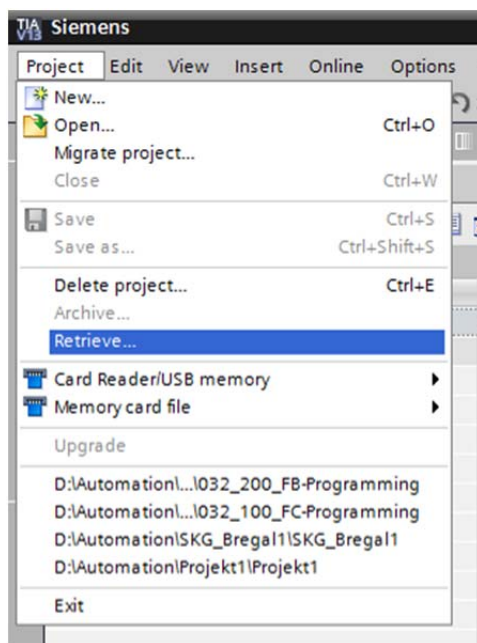
→ Before we begin with diagnostics via the web server, we need a project from the SCE_EN_032-410 Basics_Diagnostics module.

(e.g., SCE_EN_032-410_Basics_Diagnostics_2_R1503.zap13)

To retrieve an existing project that has been archived, you must select the relevant archive with →Project →Retrieve in the project view.

Confirm your selection with "Open".

(→ Project → Retrieve → Select a .zap archive → Open)

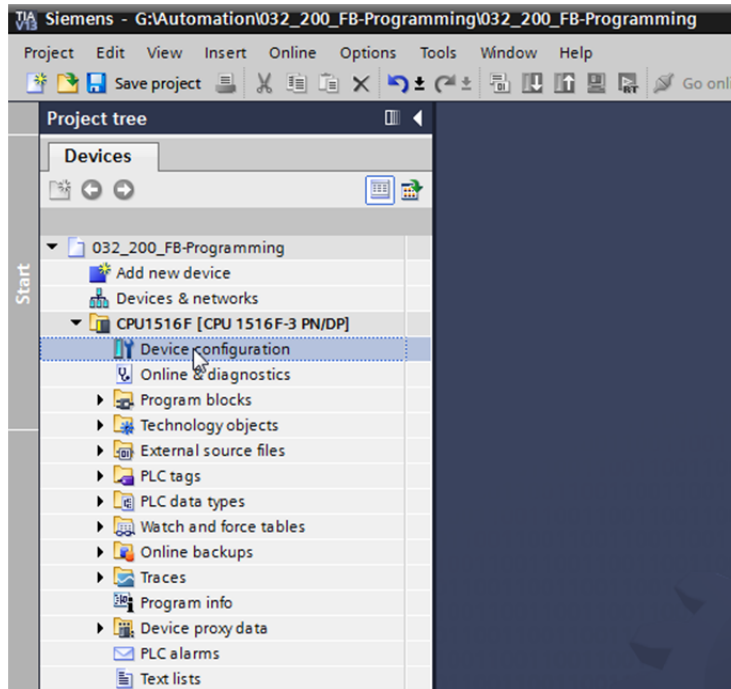


→ The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK".

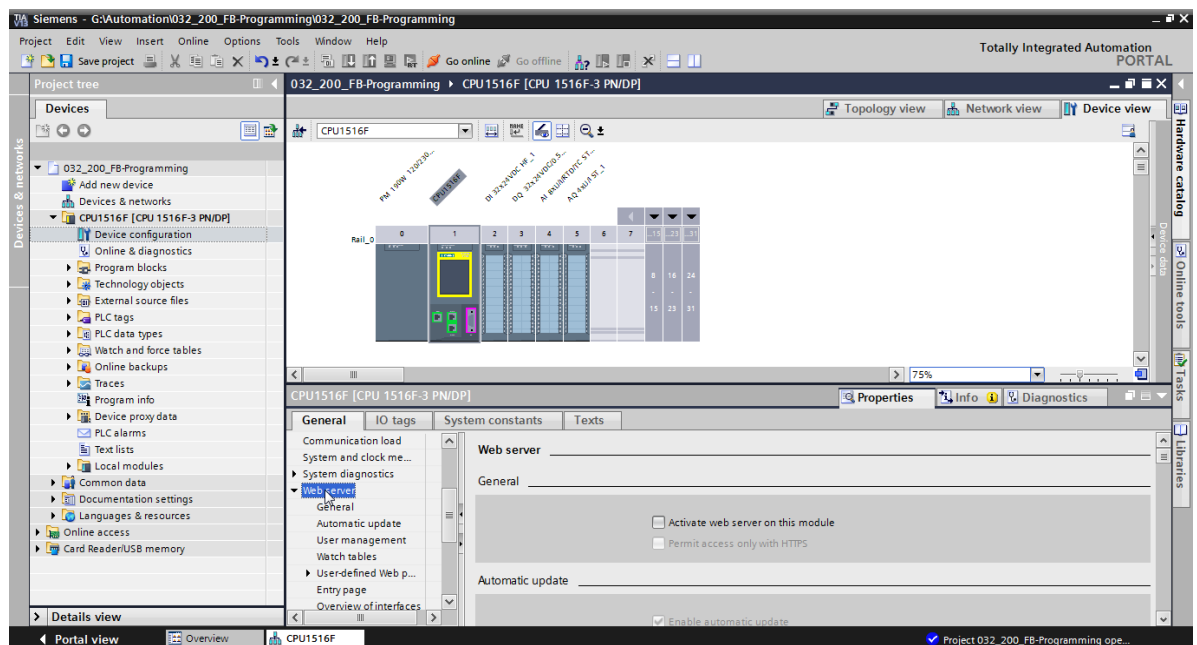
(→ Target directory → OK)

6.2 Configure the web server

- To configure the web server, open the device configuration of the CPU 1516F-3 PN/DP. (→ CPU_1516F [CPU 1516F-3 PN/DP] → Device configuration)

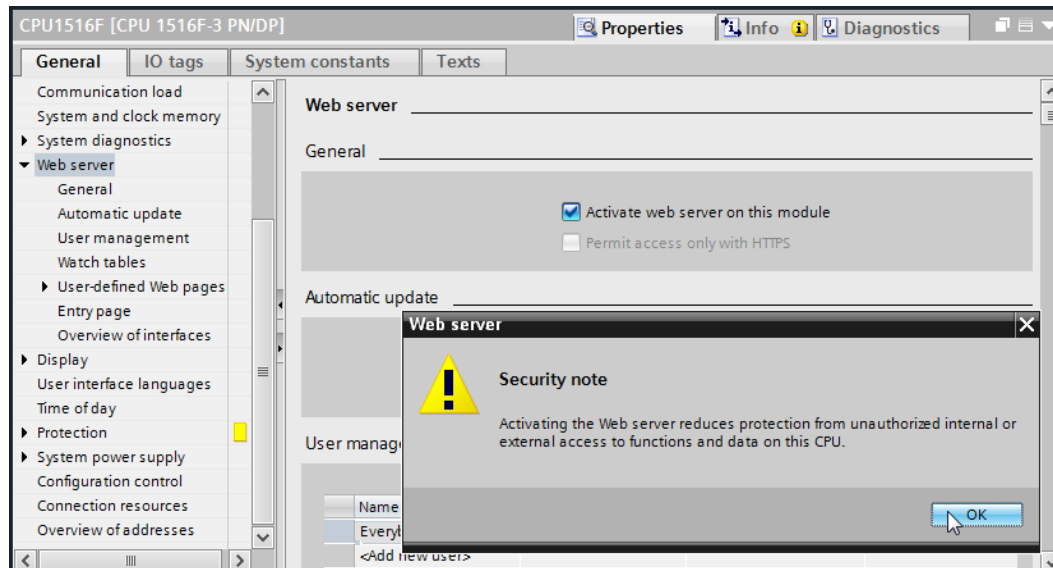


- Select the CPU and choose the 'Web server' menu item in the properties.
(→ CPU_1516F → Properties → Web server)



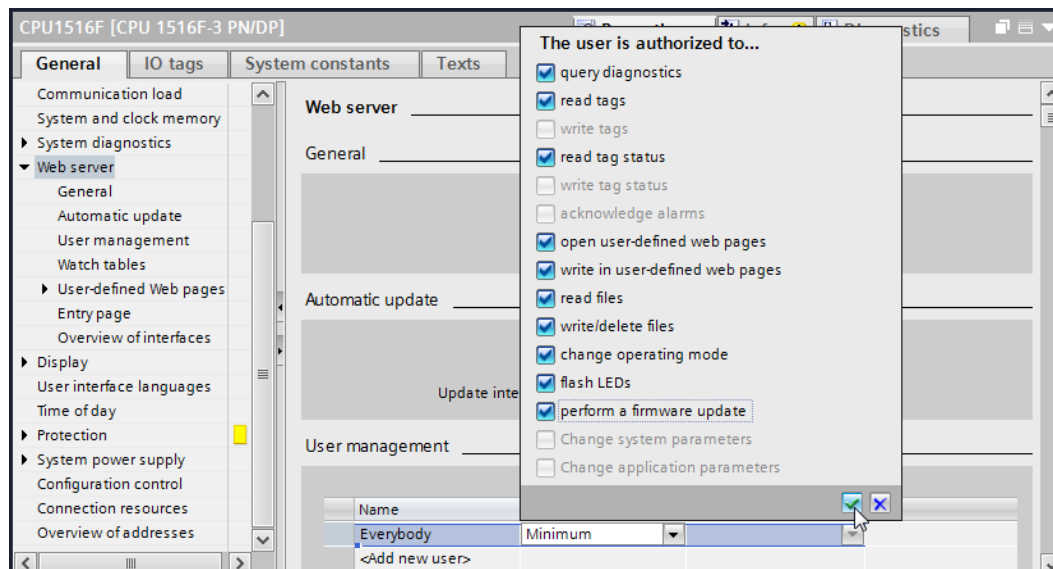
→ Activate the web server on this module and confirm the security note.

(→ ☒ Activate web server on this module → OK)



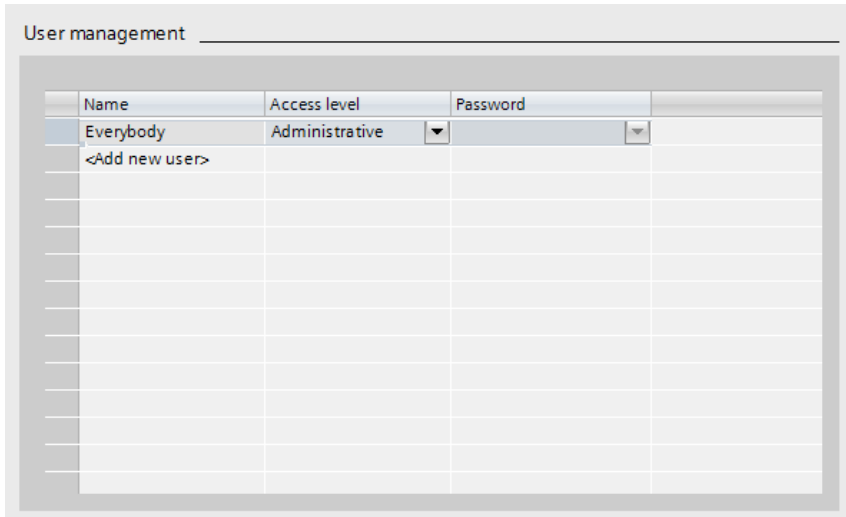
→ Leave the check mark ☒ for 'Enable automatic update', and select the security settings of the 'Everybody' user. Enable this user to carry out all possible actions and accept your settings.

(→ ☒ ☒ ☒ ☒ ☒ ☒ ☒ ☒ ☒ ☒ → ☒)




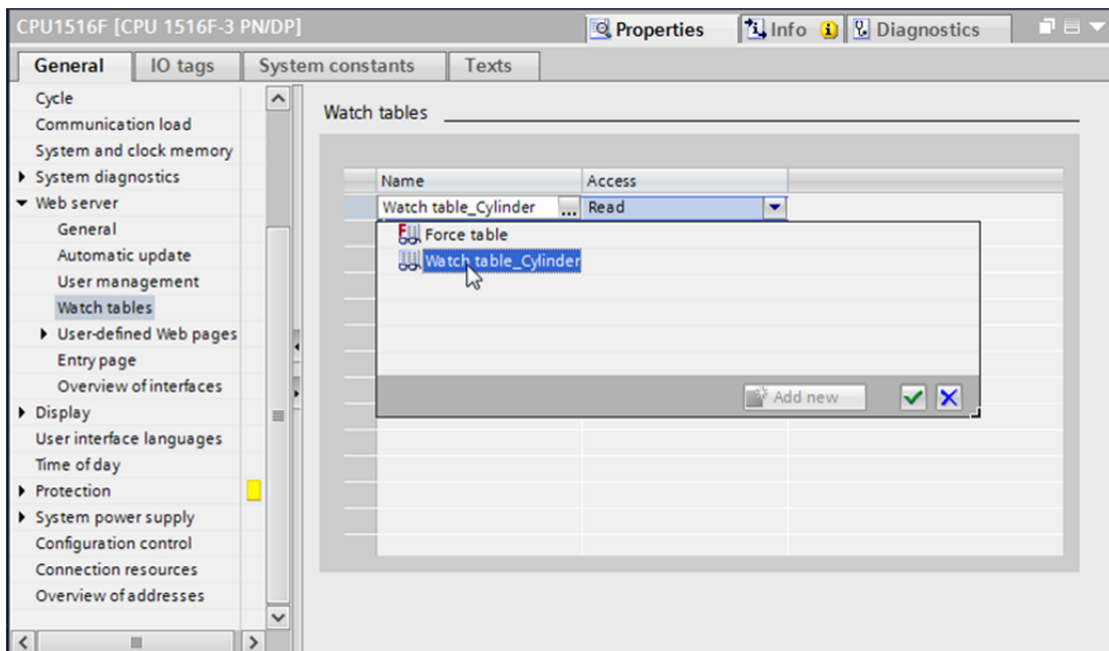
Notes: You can also create multiple users here with different authorizations. These users then require a password.

- As a result of these authorizations, the 'Everybody' user is now automatically assigned the access level 'Administrative'.

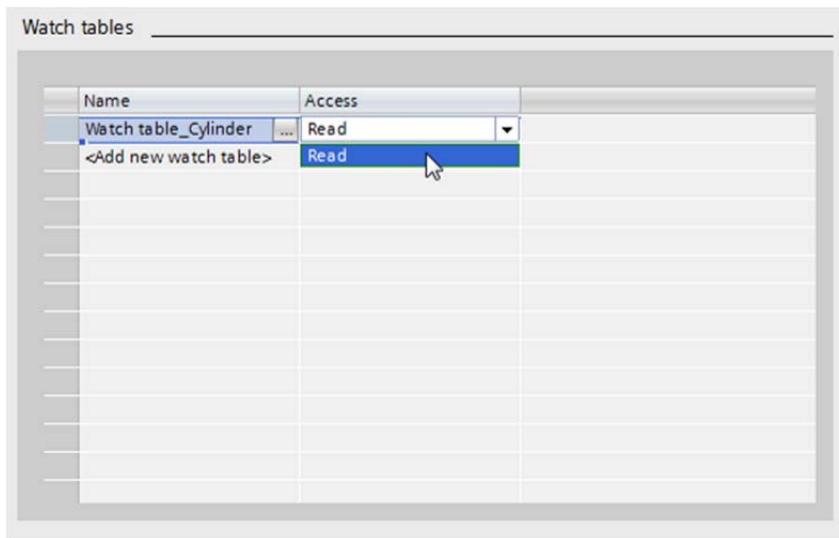


- In the 'Watch tables' menu item, the 'Watch table_Cylinder' can now be entered in the web server.

(→Watch table_Cylinder → )

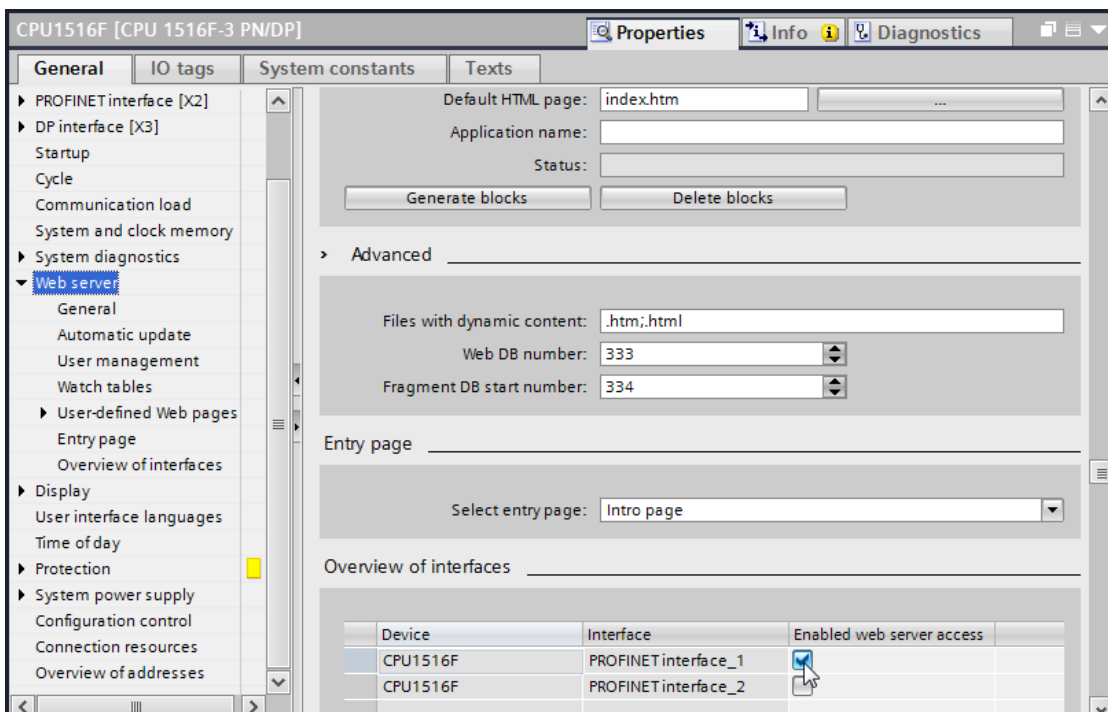


→ Accessing is read-only. (→ Read)



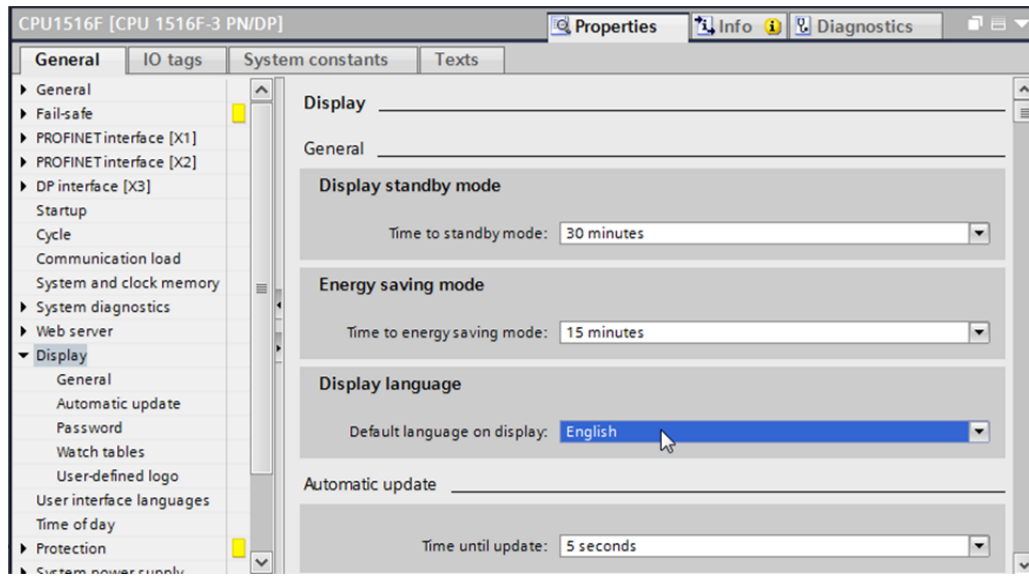
→ User-defined web pages will not be created here. For reasons of plant safety / security, we will enable only PROFINET interface_1 for access to the web server.


(→ Enabled web server access → ☒ PROFINET interface_1)

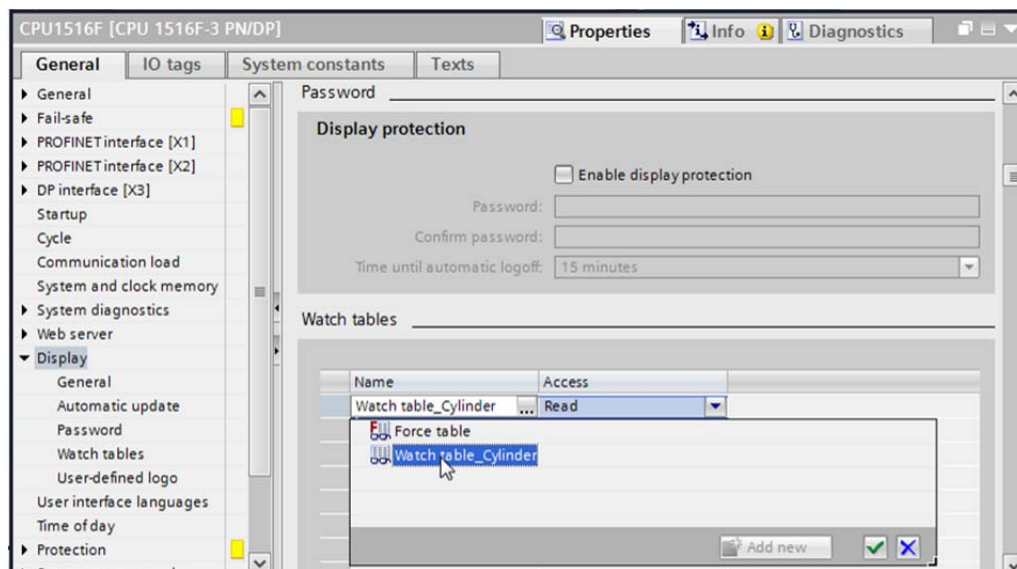


6.3 Configure the display

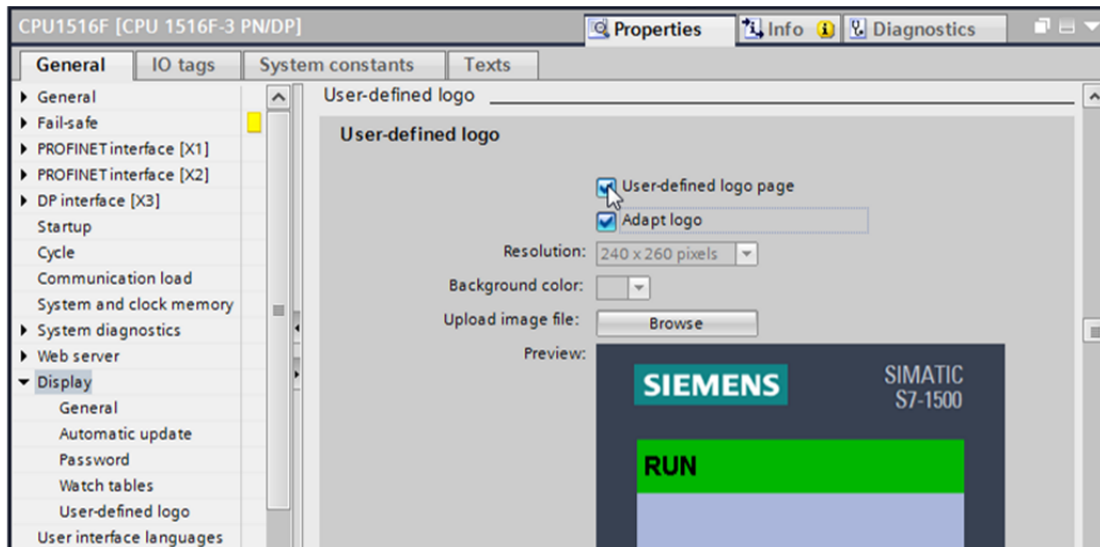
- The settings for the display of diagnostics data can also be changed on the integrated display of the CPU 1516F-3 PN/DP. First, the general settings are selected as shown here.
(→ Display → General)



- In the 'Watch tables' menu item, the 'Watch table_Cylinder' can now be entered in the display.
(→ Watch table_Cylinder → )

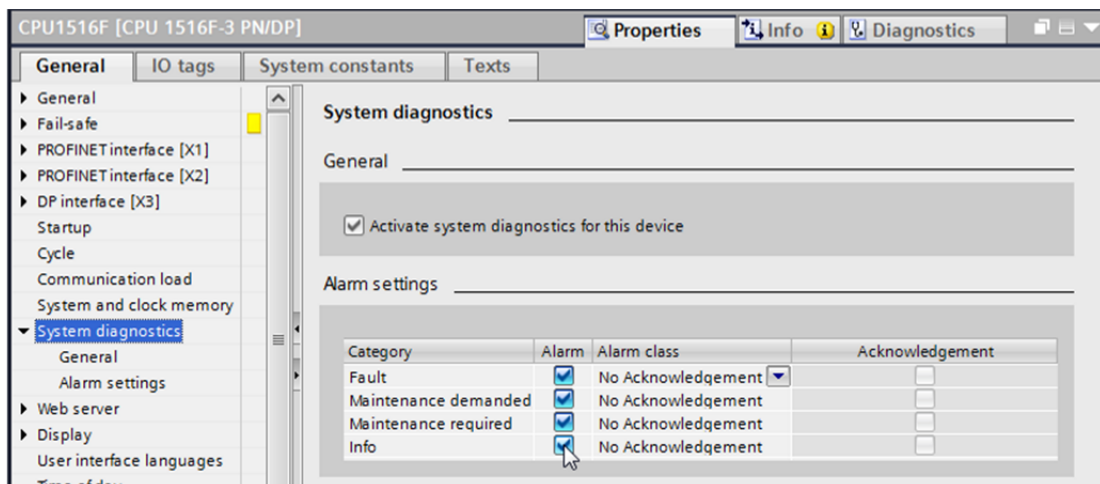


- If desired, a user-defined logo can also be shown on the display
 (→ User-defined logo page)




6.4 Configure system diagnostics

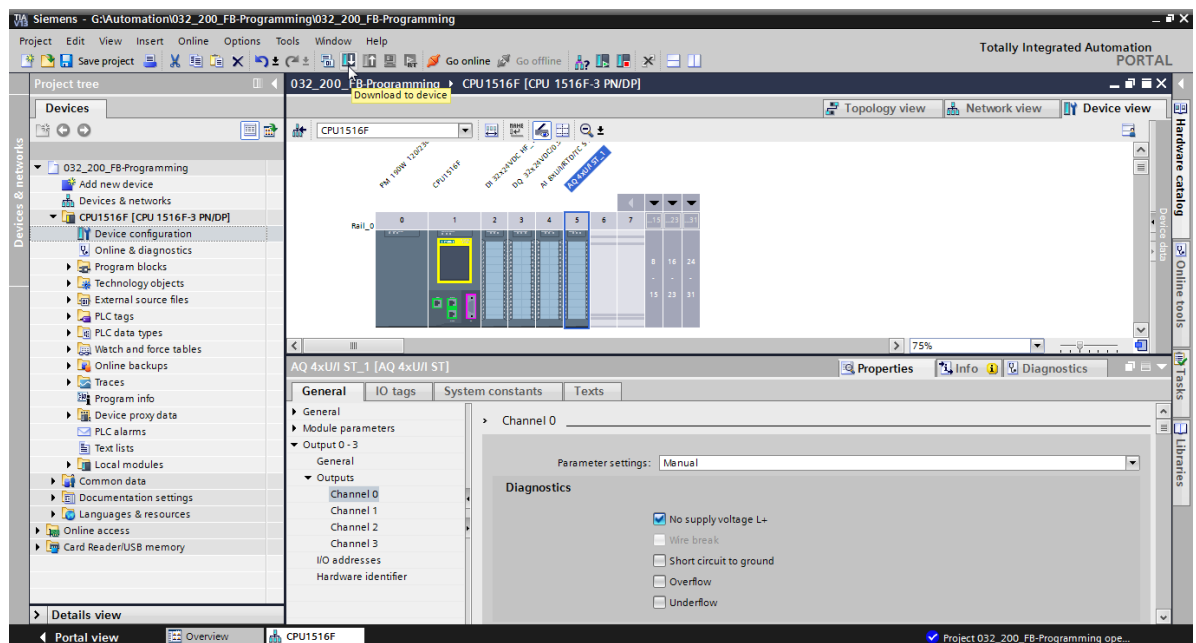
- An important function for effective troubleshooting is the integrated system diagnostics. This is always activated for the SIMATIC S7-1500. The alarm categories can be selected in the alarm settings and, if desired, an 'Acknowledgment' can be specified.



Notes: The indicated alarm class is important so that it can be selected in the alarm windows of the operator panel (e.g., TP1500, TP700, etc.).

6.5 Activate the diagnostics of the power supply for the anal. output module/download the PLC

- Once the web server, display and system diagnostics have been configured in the controller, we also activate the diagnostics for the supply voltage for the analog output module. The controller can then be selected and downloaded together with the created program.
- (→ Device configuration → AQ 4xU/I ST_1 → Output 0 – 3 → Outputs → Channel 0 → Diagnostics → ☒ No supply voltage L+ → CPU_1516F [CPU 1516F-3 PN/DP] → )

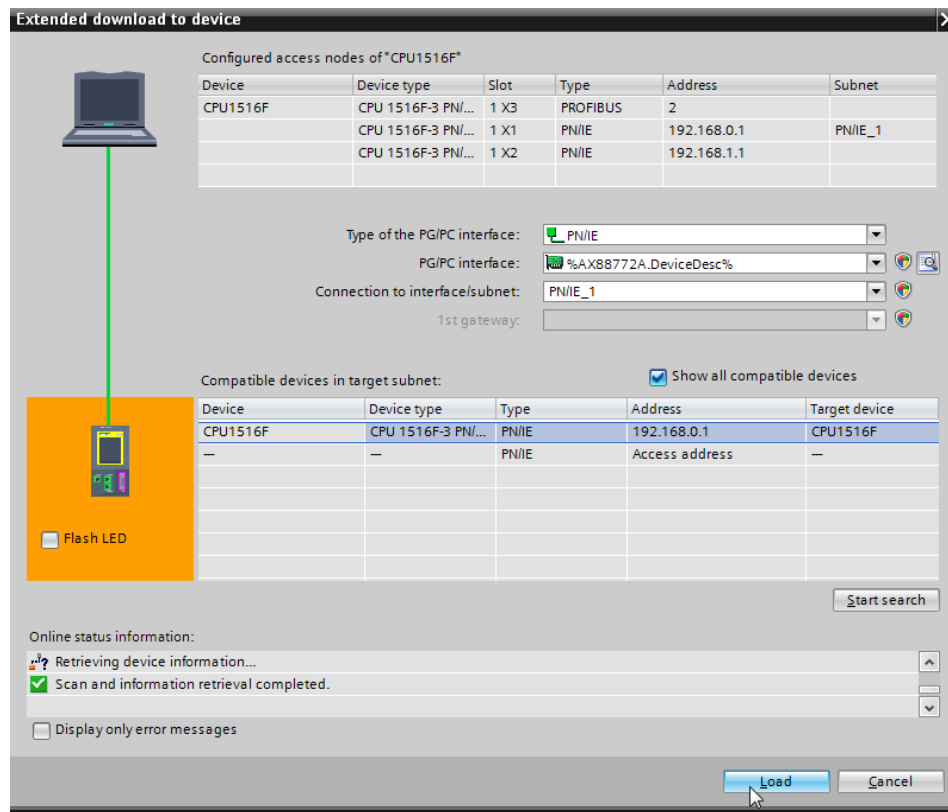


→ Select the correct interface and click 'Start search'.

(→ PN/IE → Selection of the network adapter of the PG/PC → Direct at slot '1 X1' → Start search)

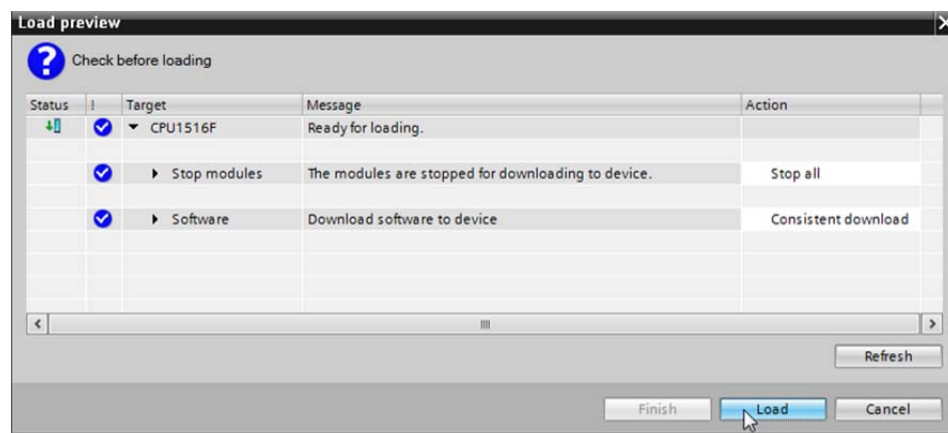
Once "Scan and information retrieval completed" appears, click 'Load'.

(→ Load)



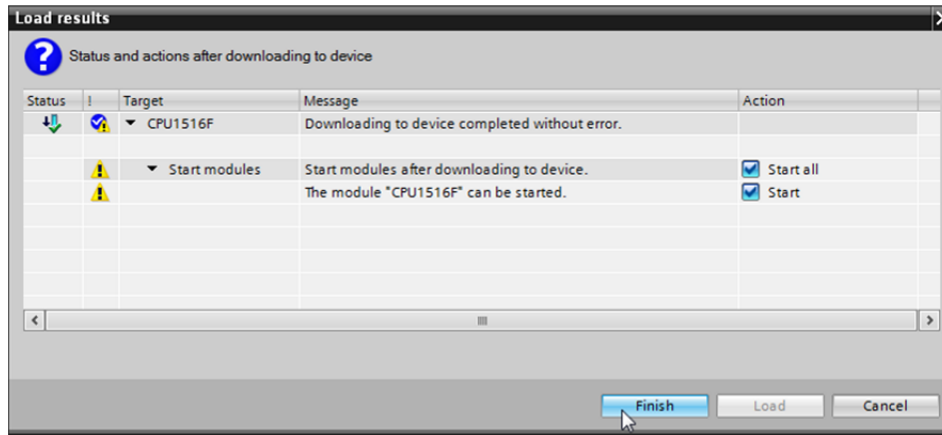
→ Before downloading can be started, other actions may have to be selected. Click 'Load' again.

(→ ☒ override all → Load)



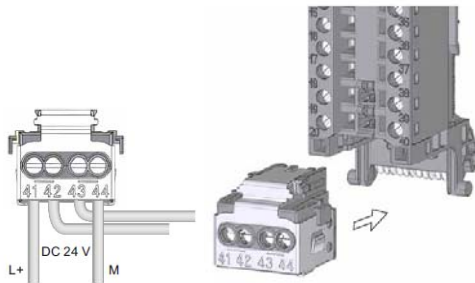
→ After loading, first select the "Start all" check box and click 'Finish'.

(→ ☒ Start all → Finish)



6.6 Trigger error message

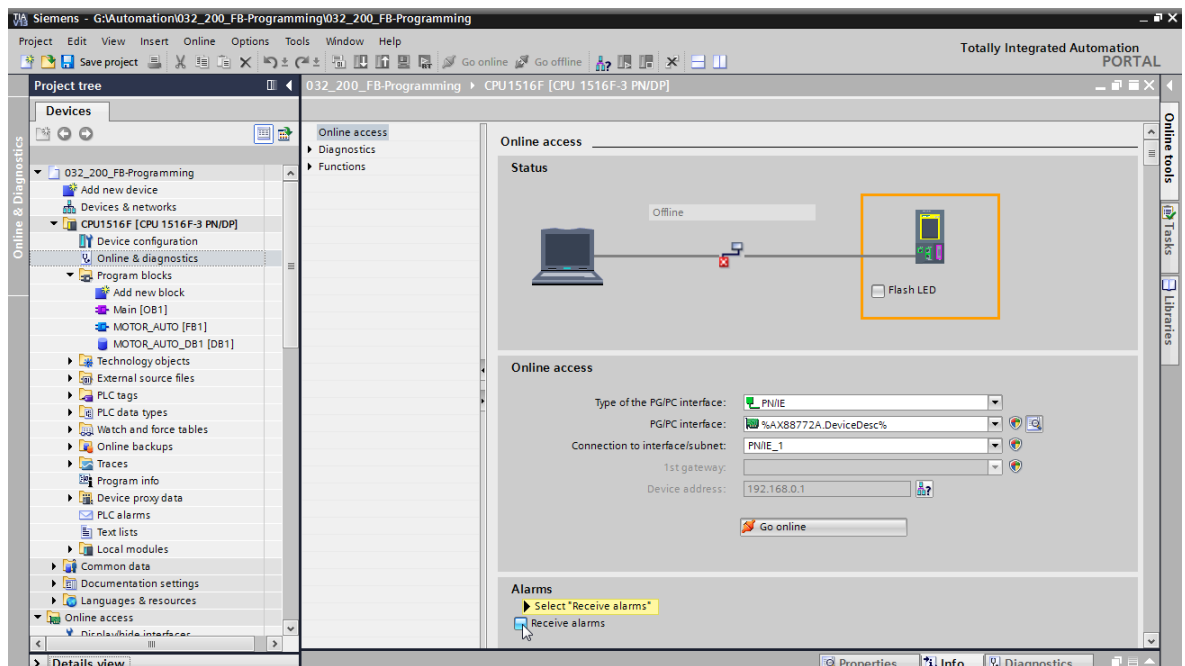
→ The power supply of the analog output module is via terminals 41-44 of the supply element. Remove this supply element, as shown here, from the front connector to trigger an error message. Result: the red ERROR LED on the CPU is lit and an error message is triggered. The following pages describe where and how you can view this error message.



6.7 Display alarms in Online & diagnostics

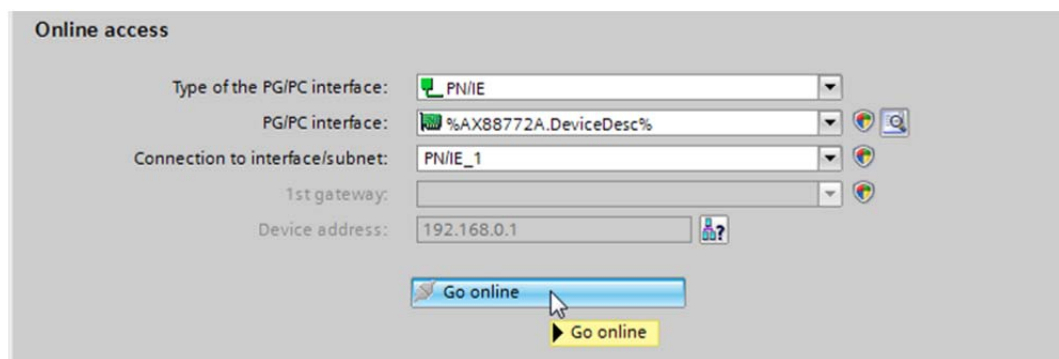
→ To get started with the diagnostic functions, we will select our controller 'CPU_1516F' and click 'Online & diagnostics'. Under 'Online access' select ☒ 'Receive alarms' for the 'Alarms' item.

(→ CPU_1516F → Online & diagnostics → Online access → Alarms → ☒ Receive alarms)

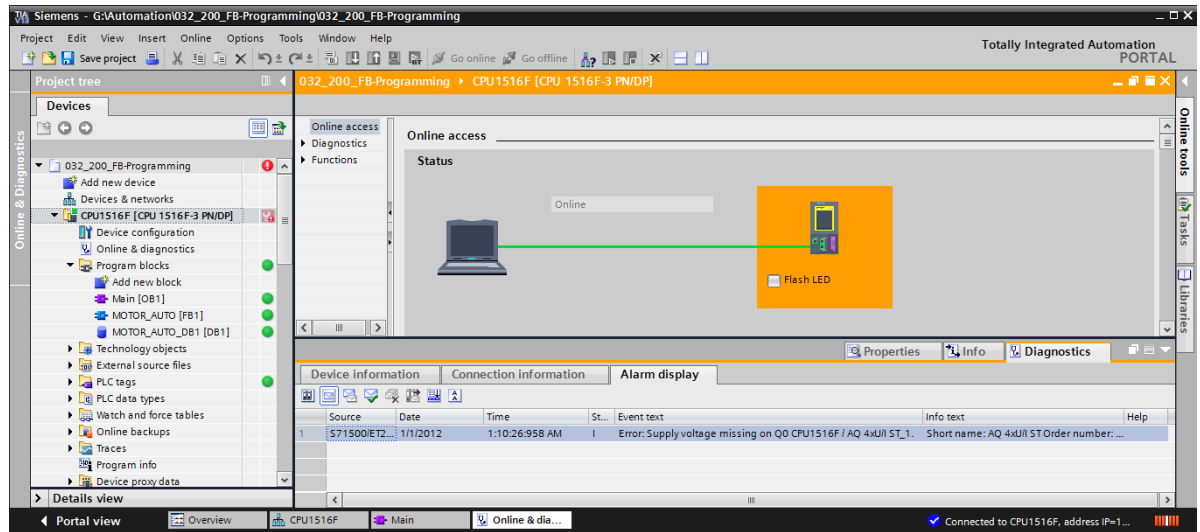


→ Select the correct interface and click 'Go online'.

(→ Go online)



- The error message can now be checked in the 'Alarm display' under 'Diagnostics'.
 (→ Diagnostics → Alarm display)

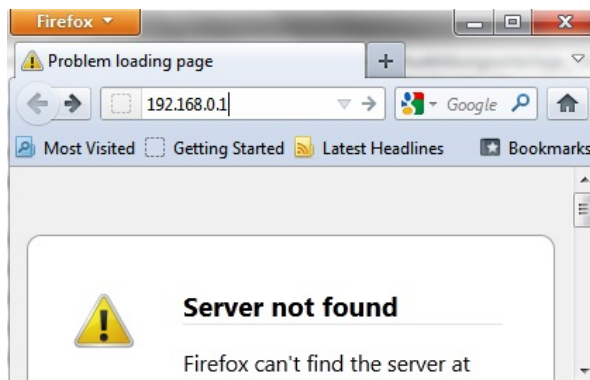


6.8 Diagnostics for the S7-1500 via the web

→ To be able to access the Web server of the CPU 315F-2 PN/DP we open any Web browser on a PC that is connected to the CPU via TCP/IP.



→ There we enter the IP address of the CPU 1516F-3 PN/DP. (→ 192.168.0.1)



- On the displayed web page, we first select the language and then click 'ENTER'.
(→ English → ENTER)



- On the 'Home Page' we see general information about the PLC and its status.
(→ Home Page)



→ Hardware, Firmware Version and Serial number are displayed besides other information under 'Diagnostics'.

(→ Diagnostics)

Diagnostics

Name [Log in](#)

- Start page
- Diagnostics**
- Diagnostic Buffer
- Module information
- Alarms
- Communication
- Topology
- Tag status
- Watch tables

Identification **Memory**

Identification:

Plant designation:

Location identifier:

Serial number: S C-F2SE01192015

Order number:

Hardware: 6ES7 516-3FN00-0AB0

Version:

Hardware: 3

Firmware: V 1.7.0

Bootloader: V 1.0.2

Diagnostics

Name [Log in](#)

- Start page
- Diagnostics**
- Diagnostic Buffer
- Module information
- Alarms
- Communication
- Topology
- Tag status
- Watch tables
- Customer pages
- Filebrowser

Identification **Memory**

Load memory

1.2% in use

23.72 MB free of 24.01 MB

Code work memory

0.0% in use

1.50 MB free of 1.50 MB

Data work memory

0.0% in use

5.00 MB free of 5.00 MB

Retentive memory

0.0% in use

472.66 KB free of 472.66 KB

SCE Training Curriculum | TIA Portal Module 032-420, Edition 02/2016 | Digital Factory, DF FA

- Under 'Diagnostics Buffer' we see descriptive information for all events in the CPU. Event information is recorded in a circular buffer. The most recent alarm is displayed in the top line.
(→ Diagnostics Buffer)

12:25:44 am 01/01/2012 English

Name

Log in

Diagnostic Buffer

Diagnostic buffer entries 1-50

Off

Start page

Diagnostics

Diagnostic Buffer

Module information

Alarms

Communication

Topology

Tag status

Watch tables

Customer pages

Number	Time	Date	State	Event
1	12:25:06.003 am	01/01/2012	incoming event	Communication initiated request: WARM RESTART Pending startup inhibit - No startup inhibit set - CPU changes from STARTUP to RUN mode
2	12:25:05.982 am	01/01/2012	incoming event	Communication initiated request: WARM RESTART Pending startup inhibit - No startup inhibit set - CPU changes from STOP to STARTUP mode
3	12:25:02.177 am	01/01/2012	incoming event	Supply voltage missing
4	12:25:01.475 am	01/01/2012	outgoing event	Supply voltage missing
5	12:25:01.389 am	01/01/2012	incoming event	Communication initiated request: STOP Pending startup inhibit(s): - No startup inhibit set - CPU changes from RUN to STOP mode
6	12:23:51.030 am	01/01/2012	incoming event	Supply voltage missing
7	12:23:46.084 am	01/01/2012	outgoing event	Supply voltage missing
8	12:19:21.717 am	01/01/2012	incoming event	Follow-on operating mode change Power-on mode set: WARM RESTART to RUN (if CPU was in RUN before) Pending startup inhibit(s): - No startup inhibit set - CPU changes from STARTUP to RUN mode Follow-on operating mode change

Details: 3

Error: Supply voltage missing on Q0 CPU1516F / AQ 4xUI ST_1.

Event ID: 16# 08:0011

incoming event

- The status of the individual modules of our SIMATIC S7-1500 is displayed with additional details in the 'Module Information' view.
(→ Module Information)

12:27:19 am 01/01/2012 English

Name

Log in

Module information

Start page

Diagnostics

Diagnostic Buffer

Module information

Alarms

Communication

Topology

Tag status

Watch tables

Customer pages

S71500/ET200MP station_1 - S71500/ET200MP station_1

Slot	State	Name	Order number	I address	Q address	Comment
1		CPU1516F	Details 6ES7 516-3FN00-0AB0			
2		DI 32x24VDC HF_1	Details 6ES7 521-1BL00-0AB0	0		
3		DQ 32x24VDC/0.5A ST_1	Details 6ES7 522-1BL00-0AB0		0	
4		AI 8xUI/RTD/TC ST_1	Details 6ES7 531-7KF00-0AB0	64		
5		AQ 4xUI ST_1	Details 6ES7 532-5HD00-0AB0		64	

State

Identification

Firmware

Error: Supply voltage missing on Q0 CPU1516F / AQ 4xUI ST_1.

- The alarm texts generated in the CPU 1516F-3 PN/DP are available in 'Alarms'.
(→ Alarms)

The screenshot shows the 'Alarms' section of the TIA Portal interface. The top bar indicates the time is 12:28:13 am on 01/01/2012, and the language is set to English. The left sidebar contains a navigation menu with options like Start page, Diagnostics, Diagnostic Buffer, Module information, Alarms (selected), Communication, Topology, Tag status, Watch tables, and Customer pages. The main area displays a table of alarms. The first alarm, with number 34, occurred on 01/01/2012 at 12:25:02.177 am. The alarm text is 'Error: Supply voltage missing on Q0 CPU1516F / AQ 4xUI ST_1'. The state is 'incoming'. Below the table, details for alarm number 34 are shown: 'Short name: AQ 4xUI ST Order number: 6ES7 532-5HD00-0AB0' and 'Incoming event'.

AlarmNr.	Date	Time	Alarm text	State	Acknowledgement
34	01/01/2012	12:25:02.177 am	Error: Supply voltage missing on Q0 CPU1516F / AQ 4xUI ST_1.	incoming	

Details on alarm number: 34
Short name: AQ 4xUI ST Order number: 6ES7 532-5HD00-0AB0
Incoming event

Note: Here we see the failure of the supply voltage for the digital input module with activated diagnostic error interrupt.

- Details about communication settings and communication errors are displayed under 'Communication'.
(→ Communication)

The screenshot shows the 'Communication' section of the TIA Portal interface. The top bar is the same as the previous screenshot. The left sidebar is the same, but 'Communication' is now selected. The main area displays the 'PROFINET Interface [X1]:' settings. It is divided into three tabs: 'Parameter' (selected), 'Statistics', and 'Resources'. Under the 'Parameter' tab, the 'Network connection' section shows the MAC address as 28-63-36-87-F3-05 and the name as cpu1516f.profinet interface_1. The 'IP parameter' section shows the IP address as 192.168.0.1, subnet mask as 255.255.255.0, and default router as ---. The 'Physical properties' section shows a table of port status.

Port number	Link status	Settings	Mode	Connection medium
X1 P1	OK	---	100 MBit/s full-duplex	Copper cable
X1 P2	disconnected	---	---	Copper cable

SCE Training Curriculum | TIA Portal Module 032-420, Edition 02/2016 | Digital Factory, DF FA

Name [Log in](#)

Communication

Parameter **Statistics** Resources Connections

Start page

Diagnostics

Diagnostics Buffer

Module information

Alarms

Communication

Topology

Tag status

Watch tables

Customer pages

Filebrowser

DataLogs

Introduction

Total statistics

Sent data packages:

Sent without errors: **3243312 Bytes**

Collision during sending attempt: **0**

Canceled due to other errors: **0**

Received data packages:

Received without errors: **755370 Bytes**

Rejected due to error: **0**

Rejected due to resource bottleneck: **0**

Statistics X1 P1

Sent data packages:

Sent without errors: **3242928 Bytes**

Collision during sending attempt: **0**

Canceled due to other errors: **0**

Received data packages:

Received without errors: **755370 Bytes**

Rejected due to error: **0**

Rejected due to resource bottleneck: **0**

Name [Log in](#)

Communication

Parameter **Statistics** **Resources** Connections

Start page

Diagnostics

Diagnostics Buffer

Module information

Alarms

Communication

Topology

Tag status




Number of connections:

Maximum connections: **256**

Connections not in use: **250**

Connections: reserved in use

ES communication	4	0
HMI communication	4	0
S7 communication	0	0
OpenUser communication	0	0
Web communication	2	6
Other communication	--	0

12:32:27 am 01/01/2012 English   

Name [Log in](#)

Communication

Parameter **Statistics** Resources **Connections**

Start page

Diagnostics

Diagnostics Buffer

Module information

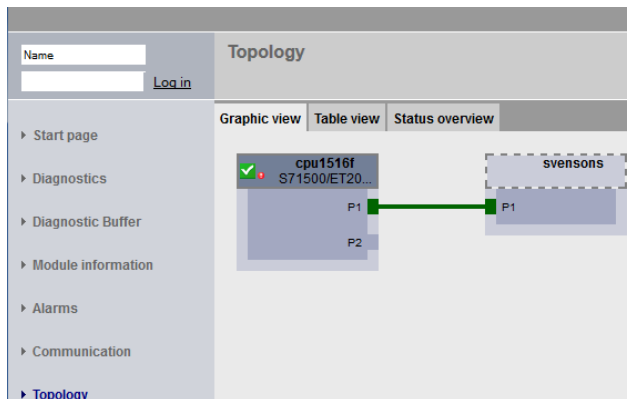
Alarms

Communication

State	Local ID (Hex)	Slot of Gateway	Remote address type	Remote address	Type	Type
✓ Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
✓ Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
✓ Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
✓ Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
✓ Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
✓ Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB

- Devices that are connected to the individual ports of the CPU 1516F-3 PN/DP and the addresses of these devices can be displayed under 'Topology'. There are various views for this. In the case of larger network structures, the entire network structure of a plant can be displayed and faulty connections shown in the status, provided this function is supported by the individual components.

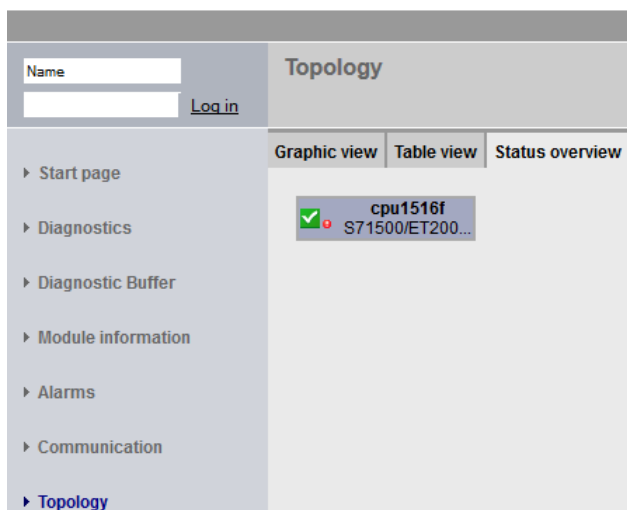
(→ Topology)



12:34:58 am 01/01/2012 English

The screenshot shows the 'Topology' view with the 'Table view' tab selected. The table displays the following data:

Port State	Name	Module type	Port	Partner port Name	Port
✓	cpu1516f	S71500/ET200MP station	port-001	svensons	port-001
?	svensons		port-002		
			port-001	cpu1516f	port-001



- Values of the individual tags can be displayed under 'Tag status'.
 (→ Tag status)

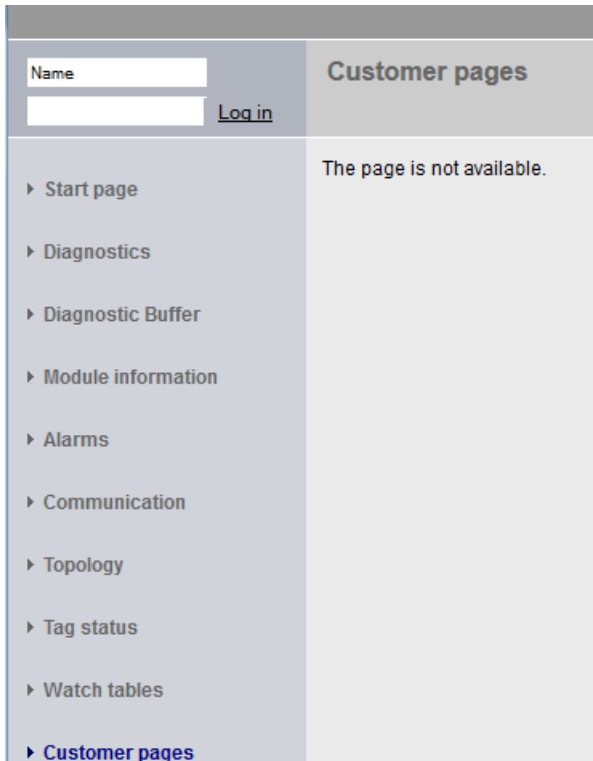
Address	Display format	Value
-K0	Bin	2#0
-A1	BOOL	FALSE
New variable		

[Apply](#)

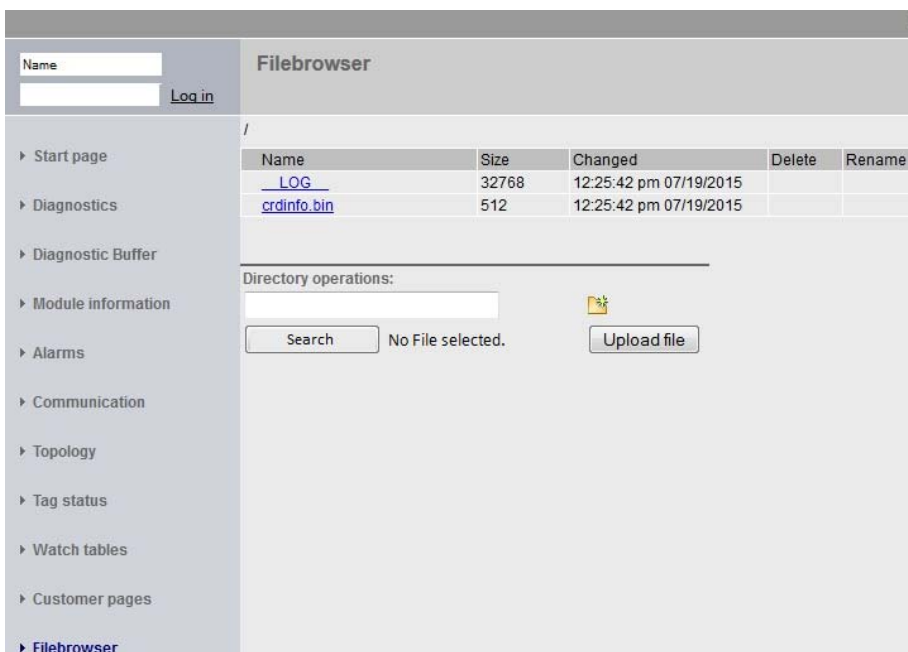
- 'Tag tables' that are linked with the web server, such as the 'Watch table_Cylinder', can also be displayed.
 (→ Tag tables → Watch table_Cylinder)

Name	Address	Format	Value
"-B1"	%E0.5	BOOL	FALSE
"-B2"	%E0.6	BOOL	FALSE
"-M2"	%A0.3	BOOL	FALSE

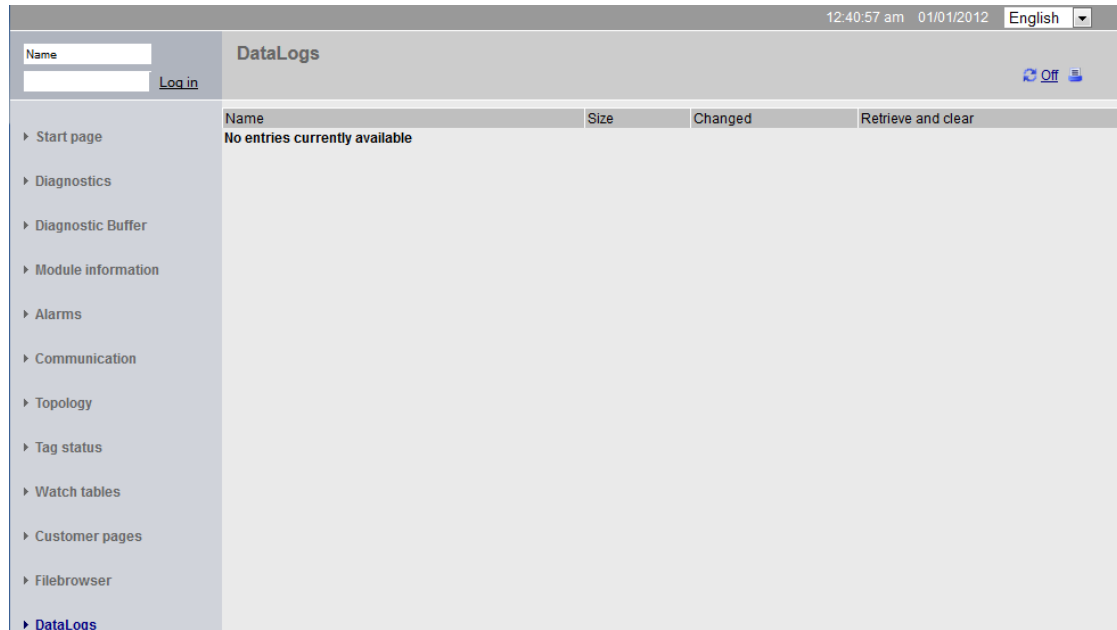
- Individually created pages for the visualization and also for operator control of processes would be seen under 'Customer pages'.
(→ Customer pages)



- Data can be stored directly on the memory card in the CPU or loaded from there using the 'Filebrowser'.
(→ Filebrowser)

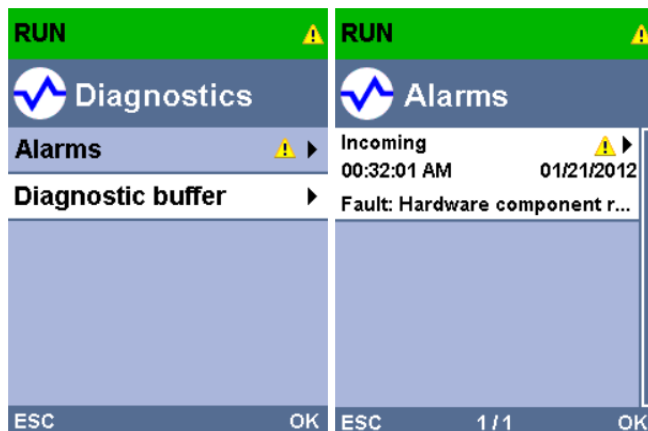


→ This means, for example, that you can read and edit the log files written by the CPU without having to use the TIA Portal. (→ DataLogs)



6.9 Diagnostics for the S7-1500 via the integrated display

→ The user also has the ability to call up a variety of diagnostic information via the display. For example, the alarm texts generated by the system diagnostics can be displayed in the 'Diagnostics' menu under 'Alarms'.
(→ Diagnostics → Alarms)



6.10 Checklist

No.	Description	Completed
1	Project 032-410_Basics_Diagnostics_2... successfully retrieved.	
2	Web server for the CPU 1516F from project 032-410_Basics Diagnostics_2... successfully configured.	
3	Display for the CPU 1516F from project 032-410_Basics Diagnostics_2... successfully configured.	
4	System diagnostics for the CPU 1516F from project 032-410_Basics Diagnostics_2... successfully configured.	
5	Diagnostics of the supply voltage for the analog output module activated.	
6	CPU 1516F from project 032-410_Basics Diagnostics_2... successfully downloaded.	
7	Power supply disconnected from analog output module.	
8	Display of alarm text from the system diagnostics in the alarm display of the TIA Portal.	
9	Display of the alarm text from the system diagnostics via the web server of the CPU 1516F.	
10	Display of the alarm text from the system diagnostics on the display of the CPU 1516F.	

7 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

[illegible]

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Please note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, please contact your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, especially Prof. Dr.-Ing. Leon Urbas und Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	4
2	Prerequisite.....	4
3	Theory.....	4
3.1	Analog signals	4
3.2	Measuring transducers	5
3.3	Analog modules – A/D converter.....	5
3.4	Data types of the SIMATIC S7-1500	6
3.5	Reading/outputting analog values	7
3.6	Normalizing analog values	8
4	Task	9
5	Planning.....	9
5.1	Analog control of the conveyor speed	9
5.2	Technology diagram	10
5.3	Reference list.....	11
6	Structured step-by-step instructions	12
6.1	Retrieve an existing project	12
6.2	Create the "MOTOR_SPEEDCONTROL" function	14
6.3	Configuration of the analog output channel.....	21
6.4	Expand the tag table to include analog signals	22
6.5	Call the block in the organization block	23
6.6	Save and compile the program.....	26
6.7	Download the program	27
6.8	Monitor program blocks	28
6.9	Archive the project.....	30
7	Checklist	31
8	Exercise	32
8.1	Task – Exercise	32
8.2	Technology diagram	33
8.3	Reference list.....	34
8.4	Planning	34
8.5	Checklist – Exercise	35
9	Additional information	36

ANALOG VALUES FOR SIMATIC S7-1500

1 Goal

In this chapter, you will become acquainted with the analog value processing of the SIMATIC S7-1500 with the TIA Portal programming tool.

The module explains the acquisition and processing of analog signals and gives a step-by-step description of read and write access to analog values in the SIMATIC S7-1500.

2 Prerequisite

This chapter builds on the chapter IEC Timers and Counters with the SIMATIC S7 CPU1516F-3 PN/DP. You can use the following project for this chapter, for example: 032-300 IEC Timers and Counters.zap13

3 Theory

3.1 Analog signals

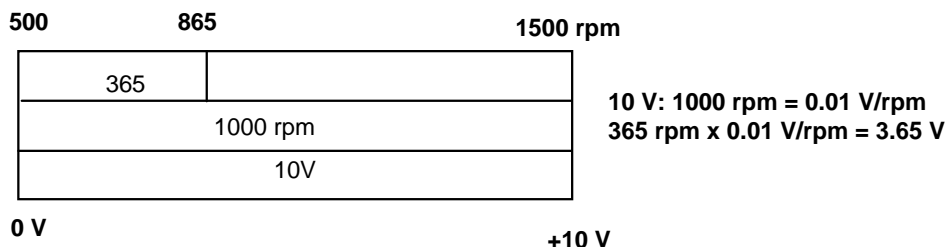
In contrast to a binary signal, which can assume only two signal states ("Voltage present +24 V" and "Voltage not present 0 V"), analog signals can assume any value within a defined range. A typical example of an analog sensor is a potentiometer. Depending on the position of the knob, any resistance can be set, up to the maximum value.

Examples of analog quantities in control engineering:

- Temperature -50 to +150 °C
- Flow rate 0 to 200 l/min
- Speed -500 to +50 rpm
- etc.

3.2 Measuring transducers

These quantities are converted to electrical voltages, currents or resistances with the help of a measuring transducer. If, for example, a speed is to be measured, the speed range of 500 to 1500 rpm can be converted to a voltage range of 0 to +10 V using a measuring transducer. At a measured speed of 865 rpm, the measuring transducer would output a voltage value of +3.65 V.

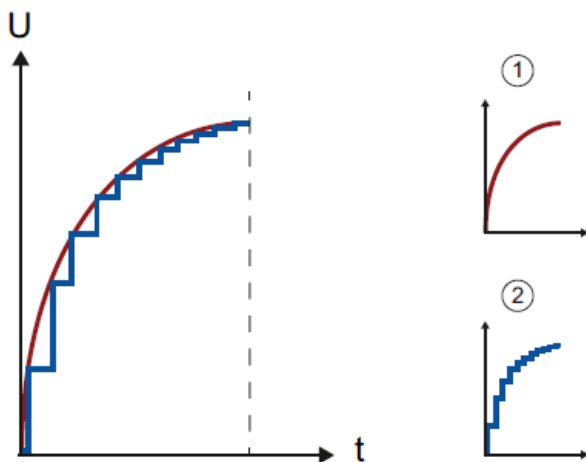


3.3 Analog modules – A/D converter

These electrical voltages, currents or resistances are then connected to an analog module that digitizes this signal for further processing in the PLC.

If analog quantities will be processed with a PLC, the read-in voltage, current or resistance value must be converted to digital information. The analog value is converted to a bit pattern. This conversion is referred to as analog-to-digital conversion (A/D conversion). This means, for example, that the voltage value of 3.65 V is stored as information in a series of binary digits.

The result of this conversion is always a 16-bit word for SIMATIC products. The integrated ADC (analog-to-digital converter) of the analog input module digitizes the analog signal being acquired and approximates its value in the form of a stepped curve. The most important parameters of an ADC are its resolution and conversion rate.

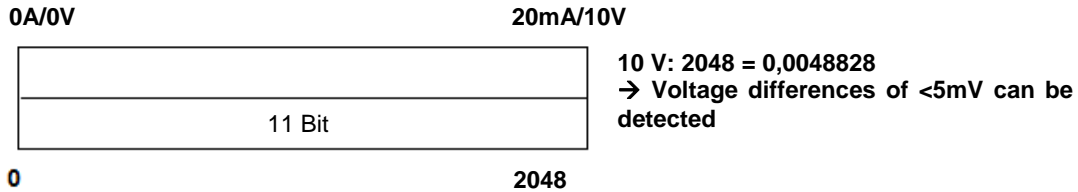


1: Analog value

2: Digital value

The more binary digits the digital representation uses, the finer the resolution is. For example, if only 1 bit was available for the voltage range of 0 to +10 V, you would only know whether the measured voltage is between 0 and +5 V or between +5 V and +10 V. With 2 bits, the range can be divided into 4 individual ranges, i.e., 0 to 2.5 / 2.5 to 5 / 5 to 7.5 / 7.5 to 10 V. Conventional A/D converters in control engineering use 8 or 11 bits for converting.

With 8 bits you have 256 individual ranges, while 11 bits provide a resolution of 2048 individual ranges.



3.4 Data types of the SIMATIC S7-1500

The SIMATIC S7-1500 has many different data types for representing different numerical formats. A list of some of the elementary data types is given below.

Data type	Size (bits)	Range	Example of constant entry
Bool	1	0 to 1	TRUE, FALSE, 0, 1
Byte	8	16#00 to 16#FF	16#12, 16#AB
Word	16	16#0000 to 16#FFFF	16#ABCD, 16#0001
DWord	32	16#00000000 to 16#FFFFFFFF	16#02468ACE
Char	8	16#00 to 16#FF	'A', 'r', '@'
Sint	8	-128 to 127	123, -123
Int	16	-32,768 to 32,767	123, -123
Dint	32	-2,147,483,648 to 2,147,483,647	123, -123
USInt	8	0 to 255	123
UInt	16	0 to 65,535	123
UDInt	32	0 to 4,294,967,295	123
Real	32	$\pm 1.18 \times 10^{-38}$ to $\pm 3.40 \times 10^{38}$	123.456, -3.4, -1.2E+12, 3.4E-3
LReal	64	$\pm 2.23 \times 10^{-308}$ to $\pm 1.79 \times 10^{308}$	12345.123456789 -1.2E+40
Time	32	T#-24d_20h_31 m_23s_648ms to T#24d_20h_31 m_23s_647ms Saved as: -2,147,483,648 ms to +2,147,483,647 ms	T#5m_30s 5#-2d T#1d_2h_15m_30x_45ms
String	Variable	0 to 254 characters in byte size	'ABC'

Note: The '**INT**' and '**REAL**' data types play a large role in analog value processing. This is because read-in analog values exist as 16-bit integers in the '**INT**' format, and in order to ensure exact further processing only '**REAL**' floating-point numbers should be used due to rounding errors in the case of '**INT**'.

3.5 Reading/outputting analog values

Analog values are read into the PLC or output from the PLC as word information. These words are accessed, for example, with the following operands:

%IW 64 Analog input word 64
 %QW 64 Analog output word 64

Each analog value ("channel") occupies one input or output word. The format is 'Int', an integer.

The addressing of input and output words conforms to the addressing in the device overview. For example:

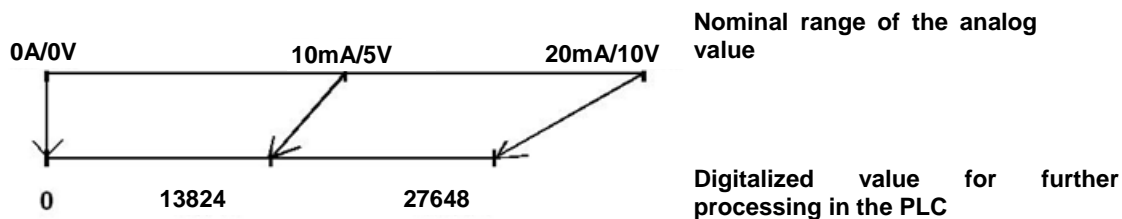
Module	Rack	Slot	I address	Q address	Type
PM 190W 120/230VAC	0	0			PM 190W 120/230...
CPU 1516F	0	1			CPU 1516F-3 PN/DP
PROFINET interface_1	0	1 X1			PROFINET interface
PROFINET interface_2	0	1 X2			PROFINET interface
DP interface_1	0	1 X3			DP interface
DI 32x24VDC HF_1	0	2	0...3		DI 32x24VDC HF
DQ 32x24VDC/0.5A ST_1	0	3		0...3	DQ 32x24VDC/0.5...
AI 8xU/I/RTD/TC ST_1	0	4	64...79		AI 8xU/I/RTD/TC ST
AQ 4xU/I ST_1	0	5		64...71	AQ 4xU/I ST

Here, the address of the first analog input would be %IW 64, that of the second analog input %IW 66, that of the third analog input %IW68, that of the fourth analog input %IW70, that of the fifth analog input %IW72, that of the sixth analog input %IW74, that of the seventh analog input %IW 76 and that of the eighth analog input %IW78.

The address of the first analog output would be %QW64, that of the second analog output %QW66, that of the third analog output %QW 68 and that of the fourth analog output %QW70.

The analog value transformation for further processing in the PLC is the same for analog inputs and analog outputs.

The digitized value ranges are as follows:



Often, these digitized values still have to be normalized by further processing them in the PLC in an appropriate manner.

3.6 Normalizing analog values

If an analog input value exists as a digitized value in the range ± 27648 , it must usually still be normalized so that the numerical values correspond to the physical quantities in the process.

Likewise, the analog output usually results from setting of a normalized value that then still has to be scaled to the output value ± 27648 .

In the TIA Portal, ready-made blocks or arithmetic operations are used for normalizing and scaling.

For this to be carried out as exactly as possible, the values for the normalizing must be converted to the REAL data type to minimize rounding errors.

4 Task

In this chapter, a function for analog control of the conveyor speed will be added to the program from chapter "SCE_EN_032-300 IEC Timers and Counters".

5 Planning

The analog control of the conveyor speed will be programmed in the "MOTOR_SPEEDCONTROL" [FC10] function as an expansion of the "SCE_EN_032-300 IEC Timers and Counters" project. This project must be retrieved from the archive in order to add this function. The "MOTOR_SPEEDCONTROL" [FC10] function will be called in the "Main" [OB1] organization block and wired. The control of the conveyor motor must be changed to - Q3 (conveyor motor -M1 variable speed).

5.1 Analog control of the conveyor speed

The speed will be set at an input of the "MOTOR_SPEEDCONTROL" [FC10] function in revolutions per minute (range: +/- 50 rpm). The data type is 32-bit floating-point number (Real).

First, the function will be checked for correct entry of the speed setpoint in the range +/- 50 rpm.

If the speed setpoint is outside the range +/- 50 rpm, the value 0 with data type 16-bit integer (Int) will be output at the output. The return value of the function (Ret_Val) will then be assigned the value TRUE (1).

If the speed setting is within the range +/- 50 rpm, this value will first be normalized to the range 0...1 and then scaled to +/- 27648 with data type 16-bit integer (Int) for output as the speed manipulated value at the analog output.

The output will then be connected with signal U1 (manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm).

5.2 Technology diagram

Here you see the technology diagram for the task.

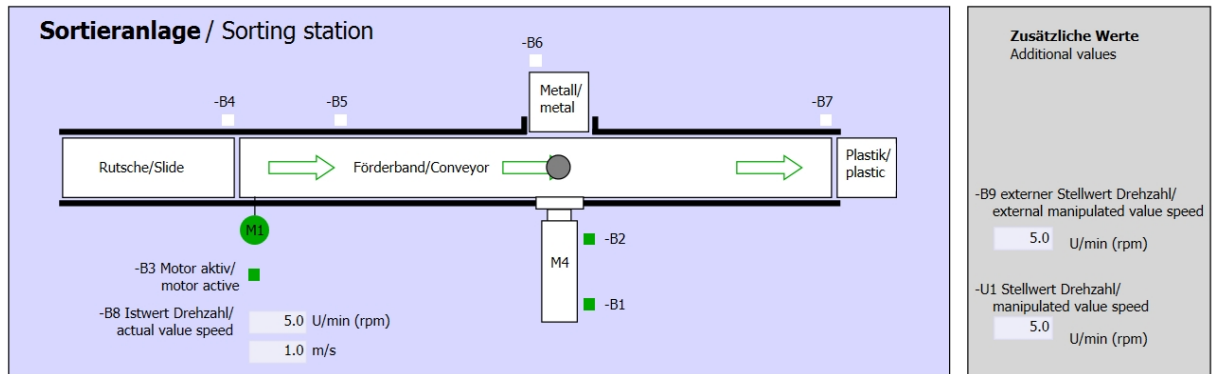


Figure 1: Technology diagram

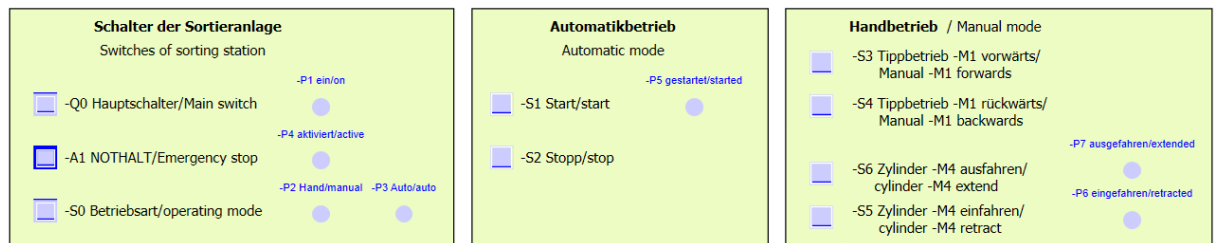


Figure 2: Control panel

5.3 Reference list

The following signals are required as global operands for this task.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop OK	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	-S1	Pushbutton automatic start	NO
I 0.4	BOOL	-S2	Pushbutton automatic stop	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO
I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO

DO	Type	Identifier	Function	
Q 0.2	BOOL	-Q3	Conveyor motor -M1 variable speed	
QW 64	BOOL	-U1	Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm	

Legend for reference list

DI Digital Input

DO Digital Output

AI Analog Input

AO Analog Output

I Input

Q Output

NC Normally Closed

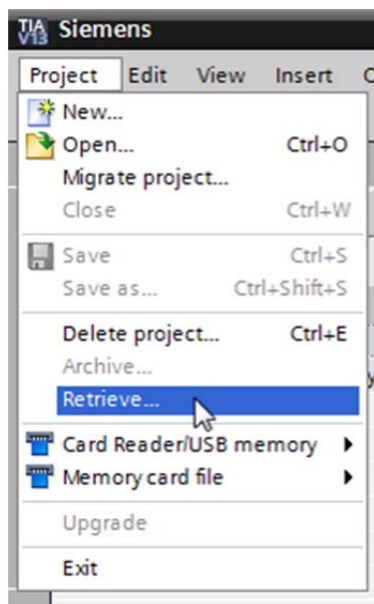
NO Normally Open

6 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

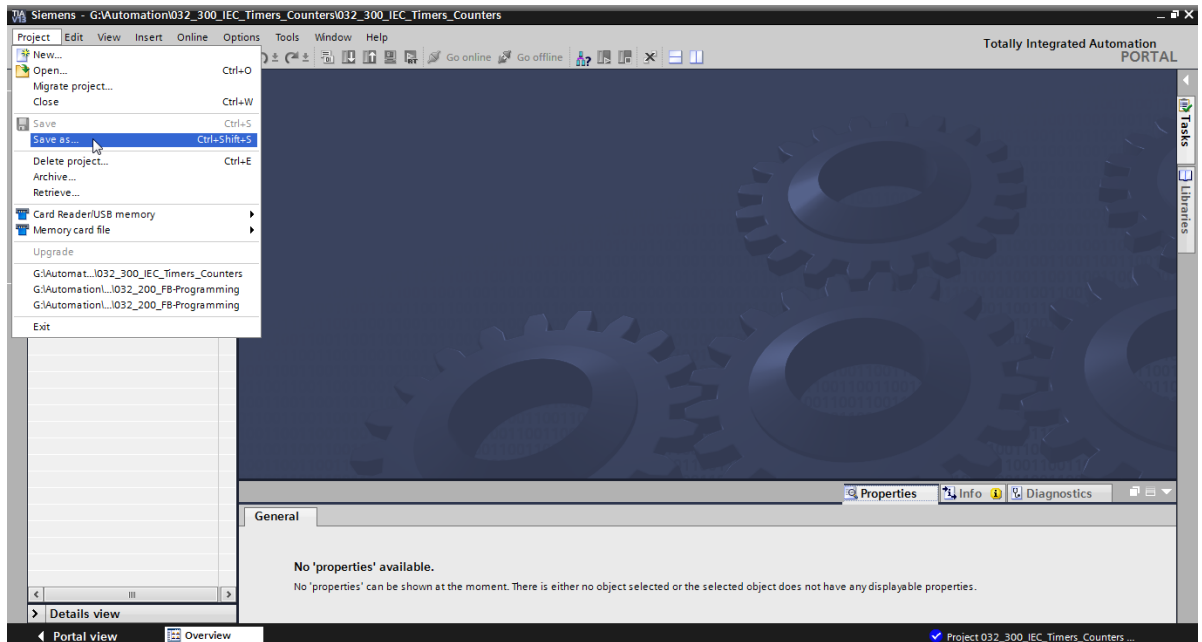
6.1 Retrieve an existing project

- Before we can expand the "SCE_EN_032-300_IEC_Timers_Counters.zap13 project from chapter "SCE_EN_032-300_IEC_Timers_Counters", we must retrieve this project from the archive. To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open.
(→ Project → Retrieve → Select a .zap archive → Open)



- The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK".
(→ Target directory → OK)

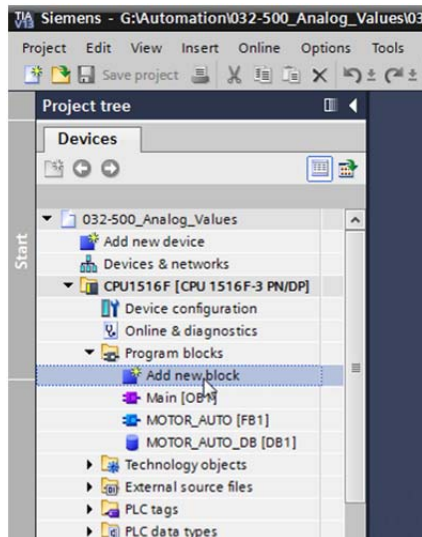
- Save the opened project under the name 032-500_Analog_Values.
(→ Project → Save as ... → 032-500_Analog_Values → Save)





6.2 Create the "MOTOR_SPEEDCONTROL" function

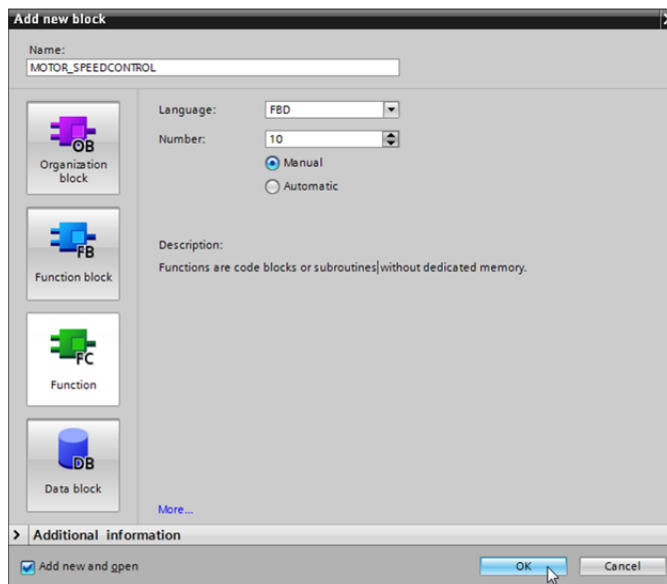
→ Select the 'Program blocks' folder of your CPU 1516F-3 PN/DP and then click "Add new block" to create a new function there.

(→ CPU_1516F [CPU 1516F-3 PN/DP] → Add new block)



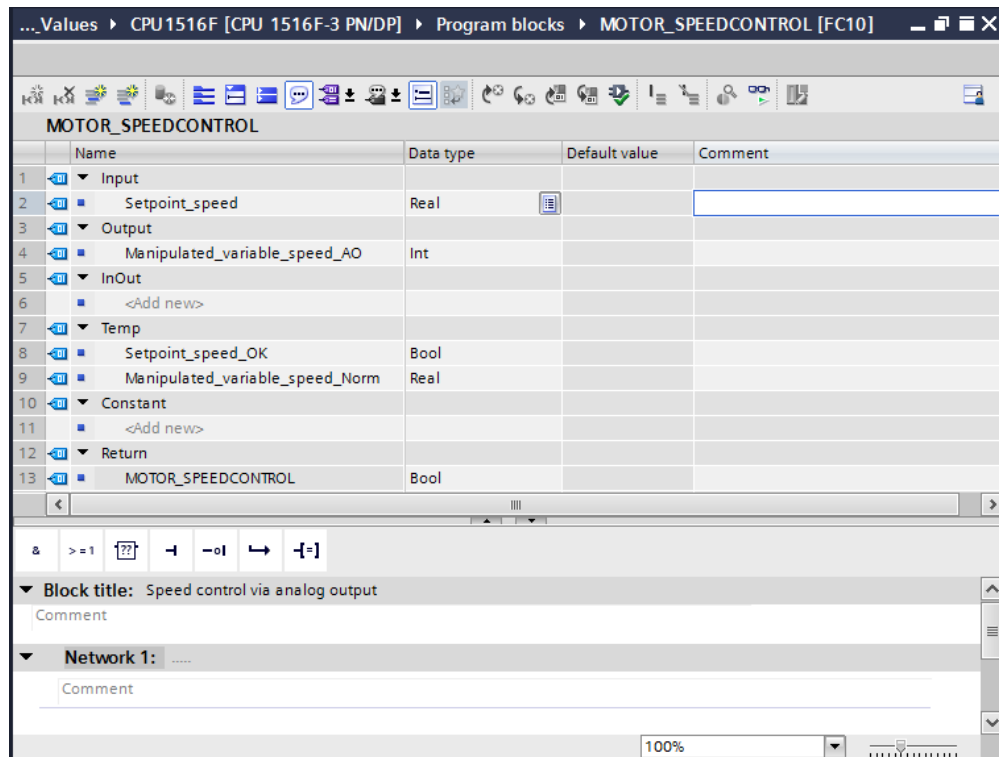
→ Select  in the next dialog and rename your new block to: "MOTOR_SPEEDCONTROL". Set the language to FBD and manually assign the number "10". Select the "Add new and open" check box. Click "OK".

(→  → Name: MOTOR_SPEEDCONTROL → Language: FBD → Number: 10 Manual → ☒ Add new and open → OK)



- Create the local tags with their comments as shown here and change the data type of the 'Return' tag from 'Void' to 'Bool'.

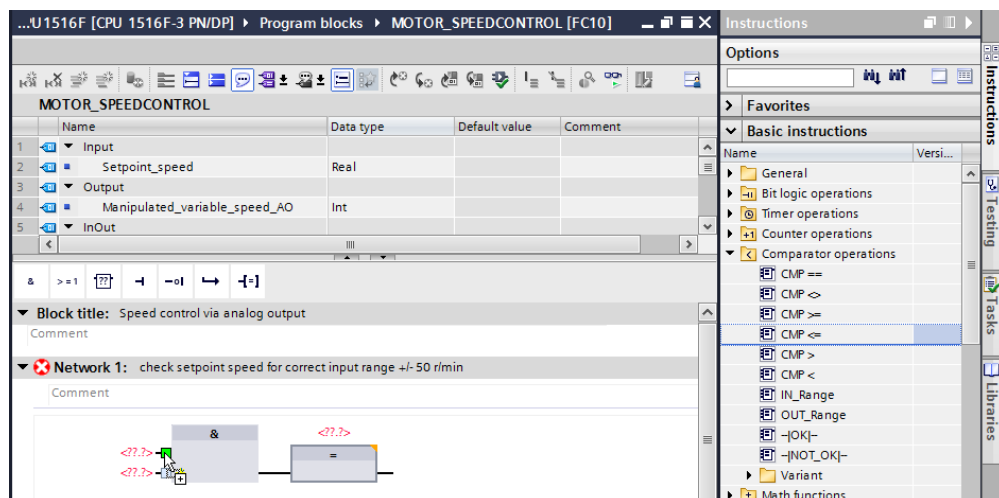
(→ Bool)



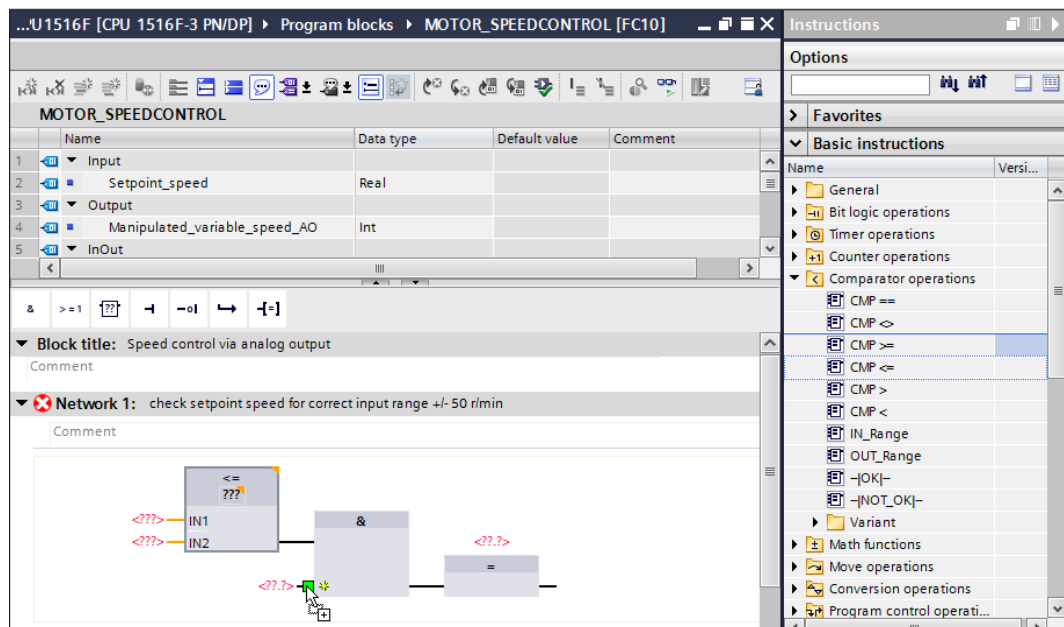
Note: Be sure to use the correct data types.

- Insert an Assignment '←=' in the first network and an 'And' '&' in front of it. Then use drag-and-drop to move the 'Comparator operation' 'Less or equal' from the 'Basic instructions' onto the first input of the '&' AND logic operation.

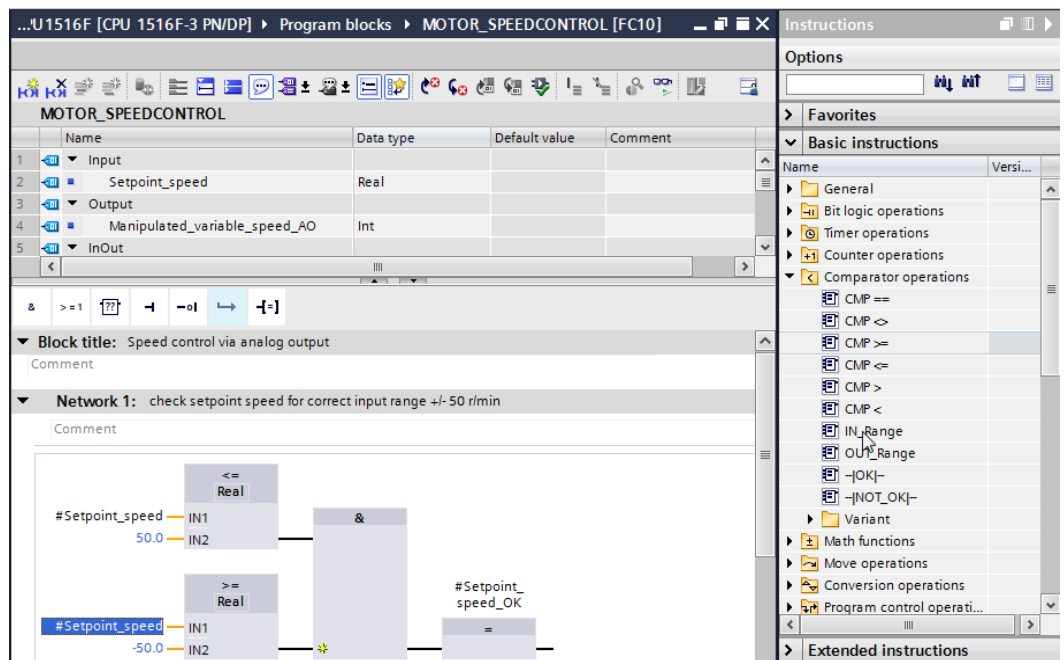
(→ ←=) → & → Basic instructions → Comparator operations → CMP<=)



- Next use drag-and-drop to move the 'Comparator operation' 'Greater or equal' onto the second input of the & AND logic operation.
- (→ Basic instructions → Comparator operations → CMP >=)

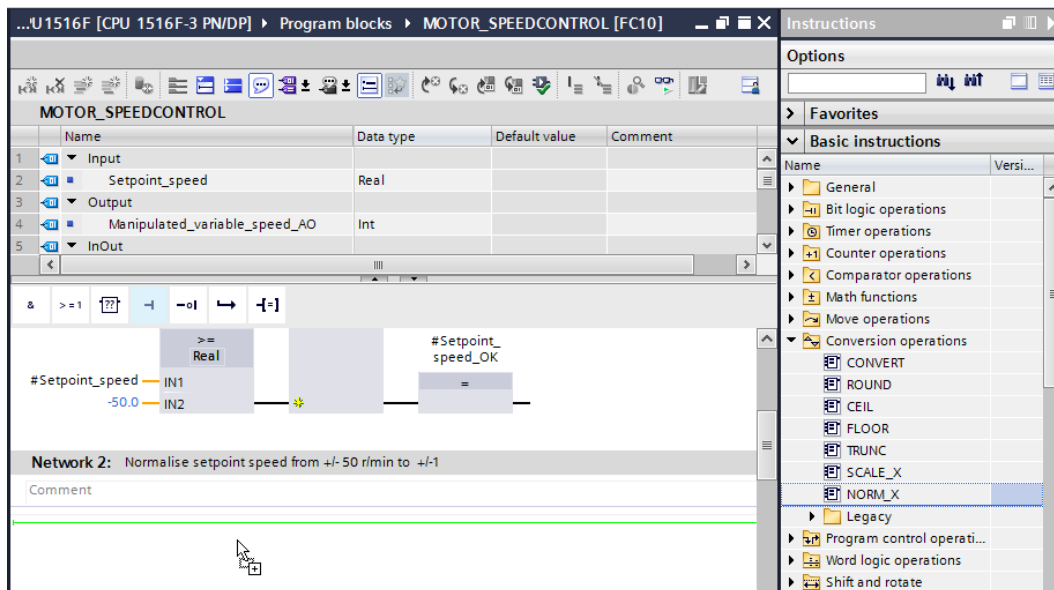


- Connect the contacts in Network 1 with the constants and local tags as shown here. The data types in the comparator operations are automatically adapted to 'Real'.

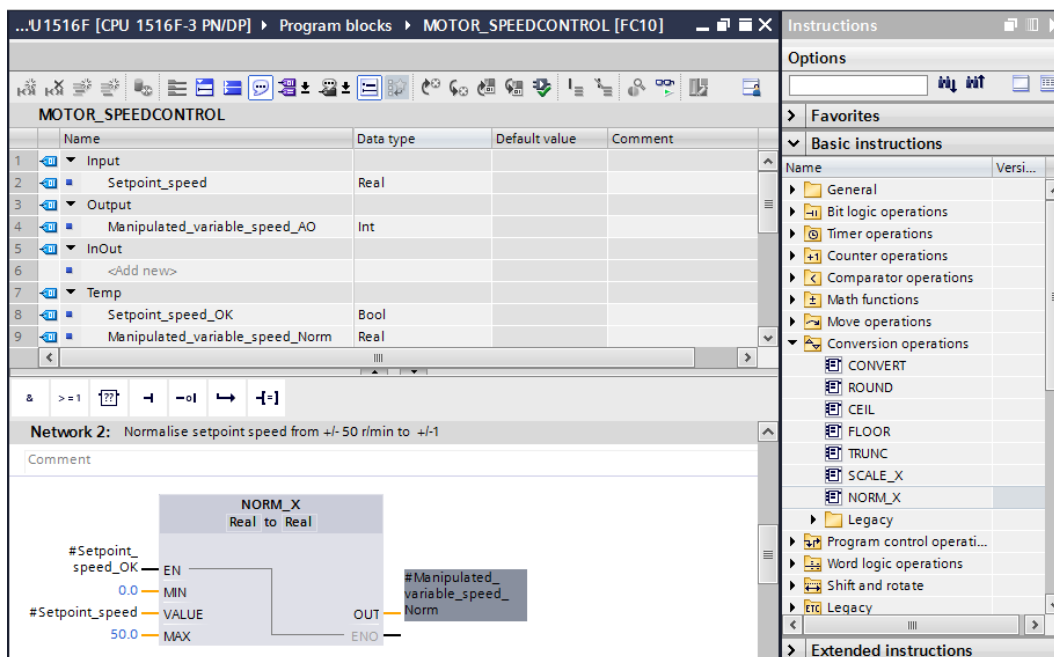


- Use drag-and-drop to move the 'Conversion operation' 'NORM_X' into Network 2 in order to normalize the speed setpoint of +/- 50 rpm to +/- 1.

(→ Basic instructions → Conversion operations → NORM_X)

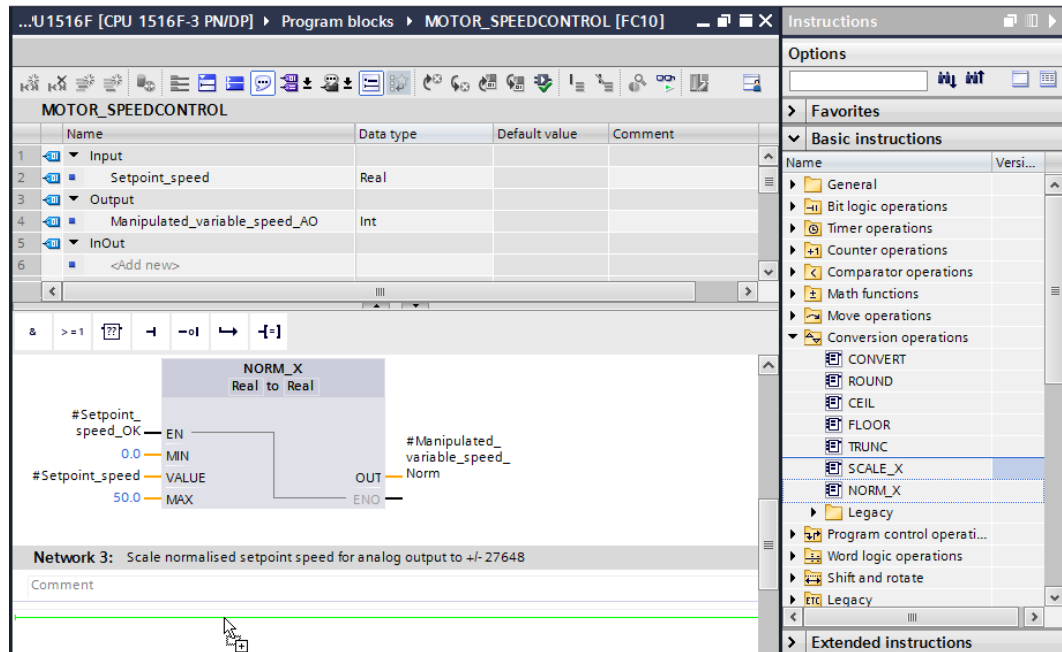


- Connect the contacts in Network 2 with the constants and local tags as shown here. The data types in 'NORM_X' are automatically adapted to 'Real'.

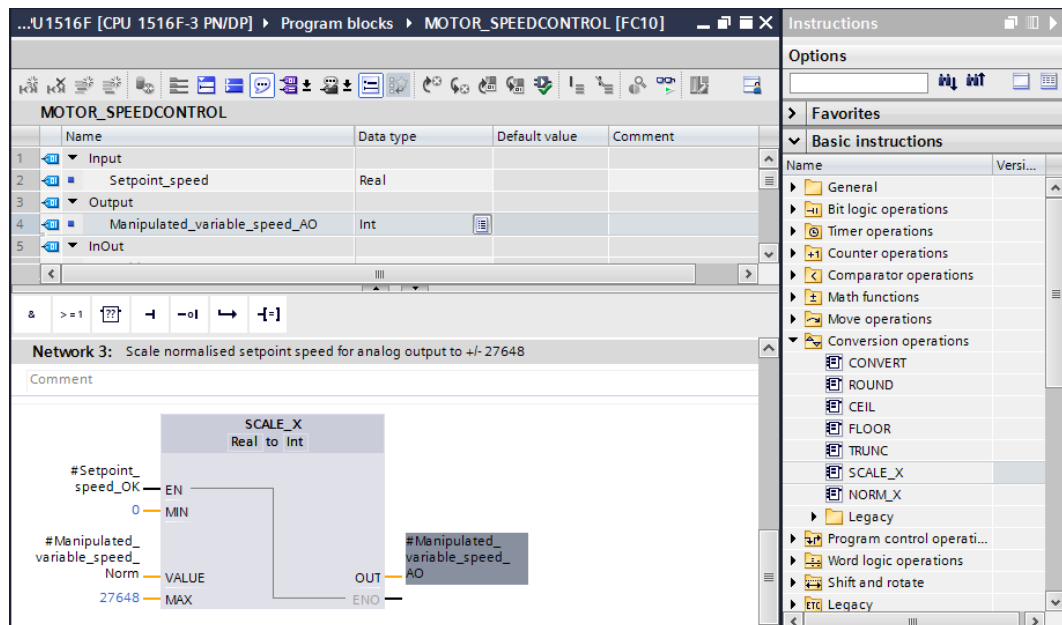



- Use drag-and-drop to move the 'Conversion operation' 'SCALE_X' into Network 3 in order to scale the speed setpoint from the normalized +/- 1 onto the range for the analog output +/- 27468.


(→ Basic instructions → Conversion operations → SCALE_X)

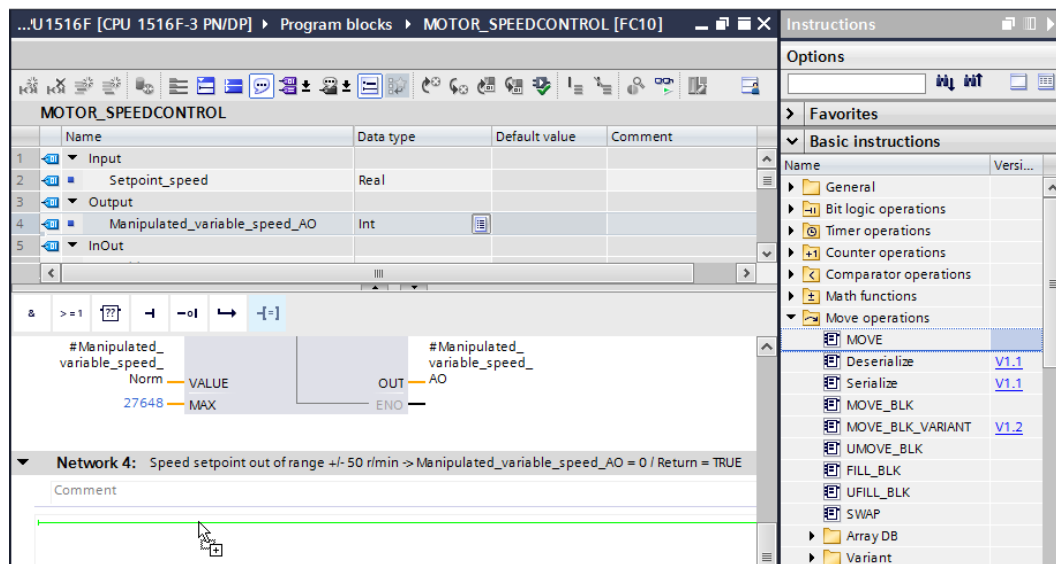


- Connect the contacts with the constants and local tags in Network 3 as well, as shown here. The data types in 'SCALE_X' are automatically changed to 'Real' or 'Int'.

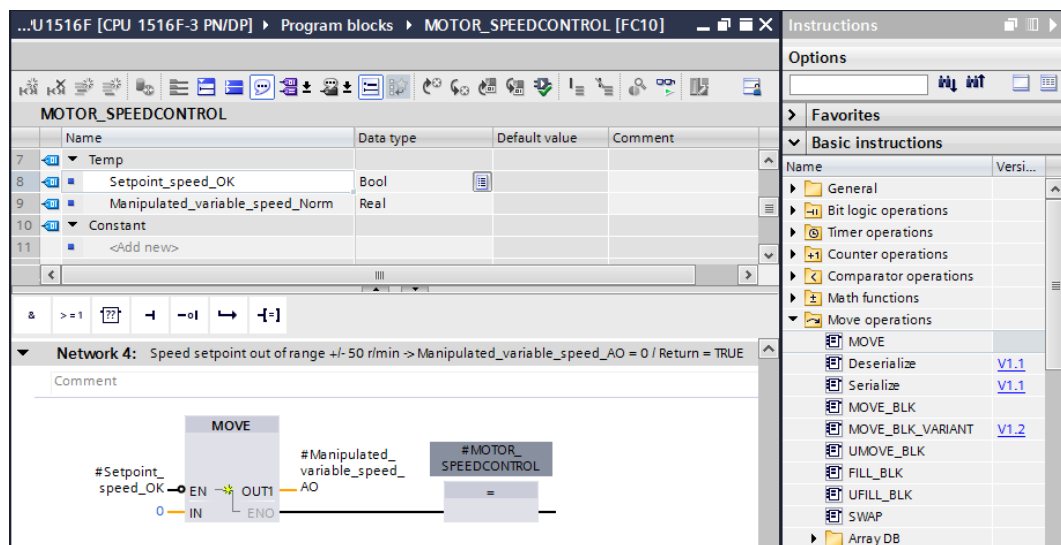


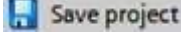
- Insert an Assignment '' in the fourth network. Use drag-and-drop to move the 'Move' command from the 'Move operations' folder under 'Basic instructions' in front of the Assignment.

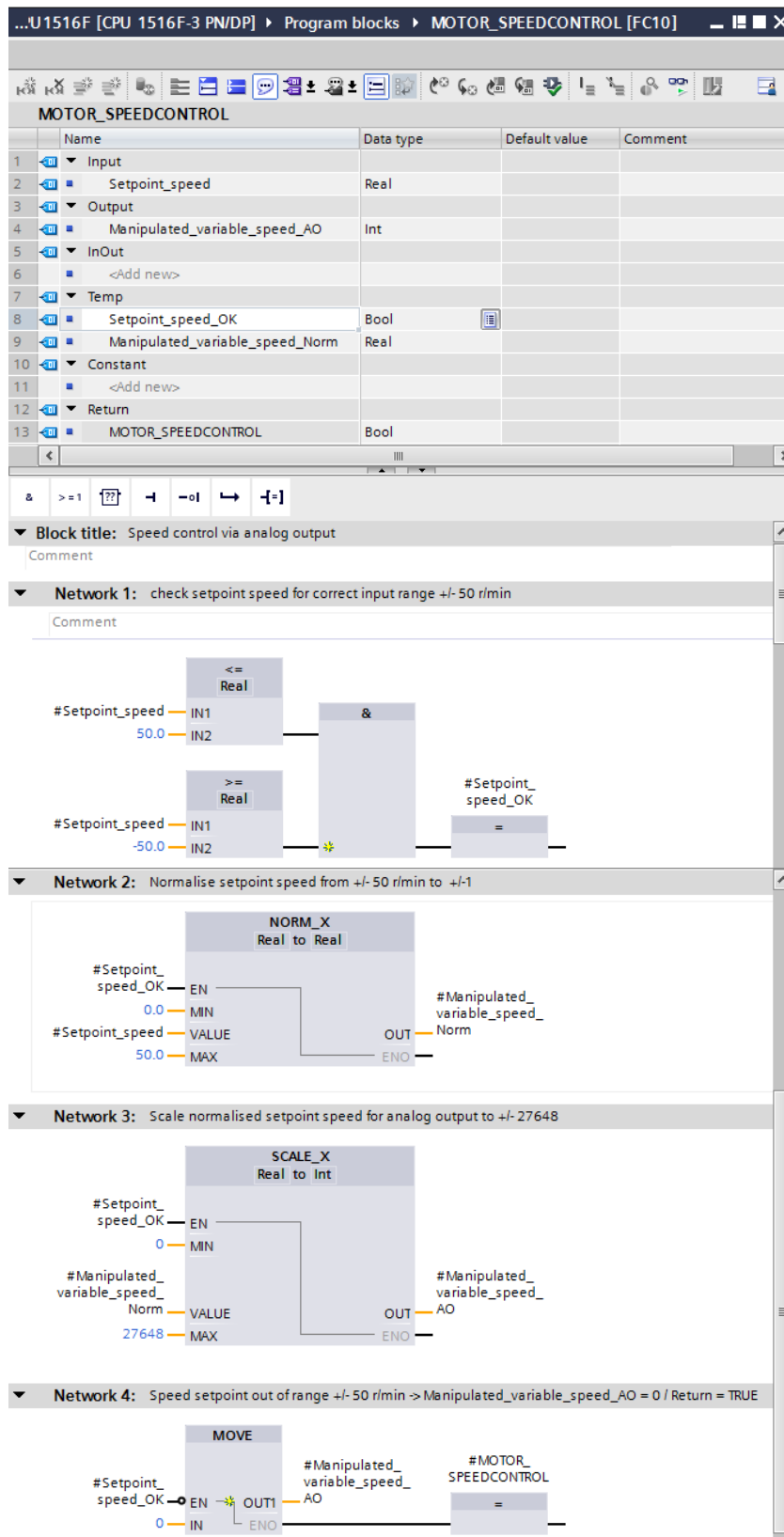
(→  → Basic instructions → Move operations → MOVE)



- The contacts in Network 4 will now be connected with constants and local tags as shown here. If the speed setpoint is not within the range +/- 50 rpm, the value '0' is output at the analog output and the value TRUE is assigned to the return value (Return) of the "MOTOR_SPEEDCONTROL" function.

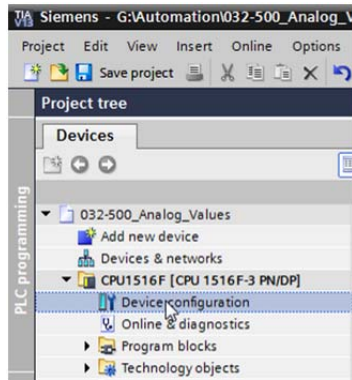


→ Do not forget to click . The finished function "MOTOR_SPEEDCONTROL" [FC10] in FBD is shown below.



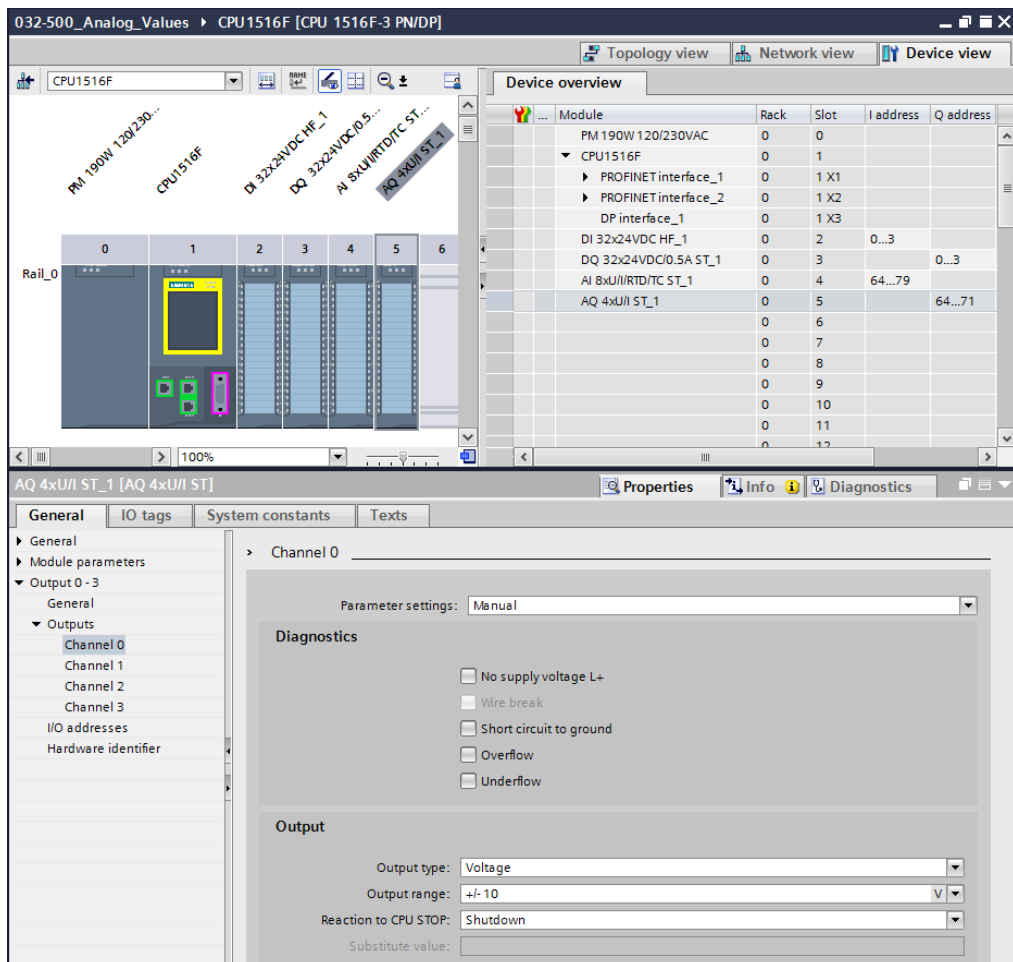
6.3 Configuration of the analog output channel

→ Double-click the 'Device configuration' to open it.



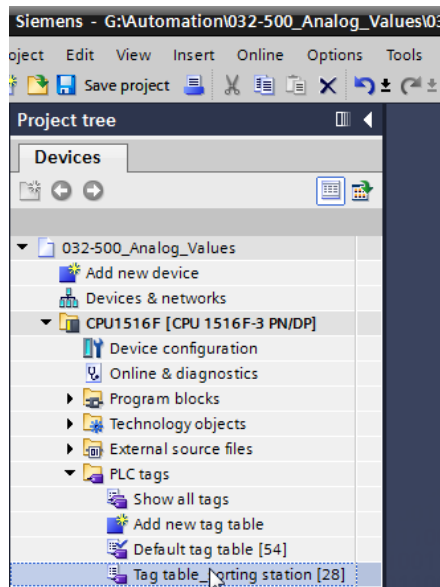
→ Check the address setting and the configuration of the analog output channel 0.

(→ Q address: 64...71 → Properties → General → Output 0 - 3 → Outputs → Channel 0 → Output type: Voltage → Output range: +/- 10 V → Reaction to CPU STOP: Shutdown)



6.4 Expand the tag table to include analog signals

→ Double-click the 'Tag table_sorting station' to open it.



→ Add the global tags for the analog value processing to the "Tag table_sorting station". An analog input B8 and an analog output U1 must be added.

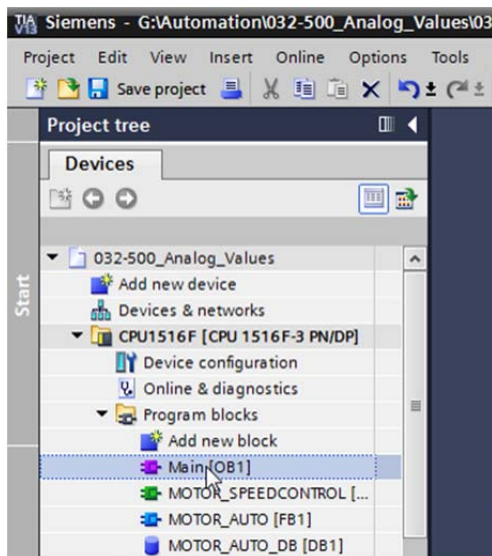
(→ U1 → %QW64 → B8 → %IW64)

032-500_Analog_Values > CPU1516F [CPU 1516F-3 PN/DP] > PLC tags > Tag table_sorting station [30]

Tag table_sorting station							
	Name	Data type	Address	Retain	Visibl...	Acces...	Comment
15	-S5	Bool	%I1.6	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode cylinder -M4 retract (no)
16	-S6	Bool	%I1.7	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode cylinder -M4 extend (no)
17	-Q1	Bool	%Q0.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 forwards fixed speed
18	-Q2	Bool	%Q0.1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 backwards fixed speed
19	-Q3	Bool	%Q0.2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 variable speed
20	-M2	Bool	%Q0.3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	cylinder -M4 retract
21	-M3	Bool	%Q0.4	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	cylinder -M4 extend
22	-P1	Bool	%Q0.5	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „main switch on“
23	-P2	Bool	%Q0.6	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „manual mode“
24	-P3	Bool	%Q0.7	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „automatic mode“
25	-P4	Bool	%Q1.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „emergency stop activated“
26	-P5	Bool	%Q1.1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „automatic mode started“
27	-P6	Bool	%Q1.2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display cylinder -M4 „retracted“
28	-P7	Bool	%Q1.3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display cylinder -M4 „extended“
29	-U1	Int	%QW64	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	manipulated value speed in 2 directions +/- 10V
30	-B8	Int	%IW64	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor actual value speed 0 ...10V
31	<Add new>			<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

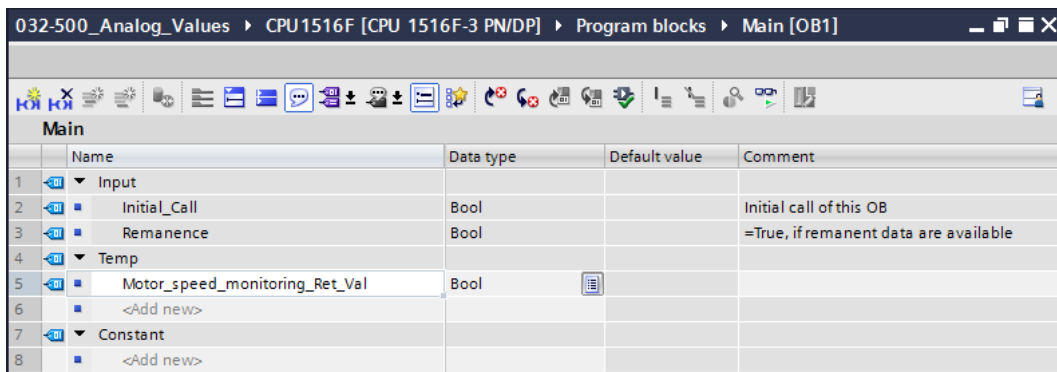
6.5 Call the block in the organization block


→ Open the "Main [OB1]" organization block with a double-click.



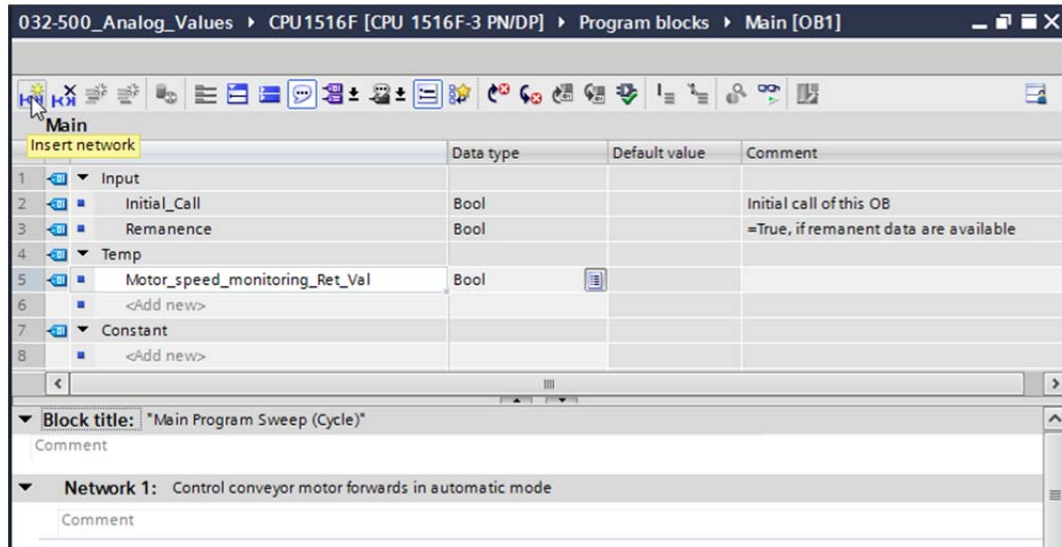
→ Add the temporary tag 'Motor_speed_monitoring_Ret_Val' to the local tags of OB1. These will be needed in order to interconnect the return value of the "MOTOR_SPEEDCONTROL" function.

(→ Temp → Motor_speed_monitoring_Ret_Val → Bool)

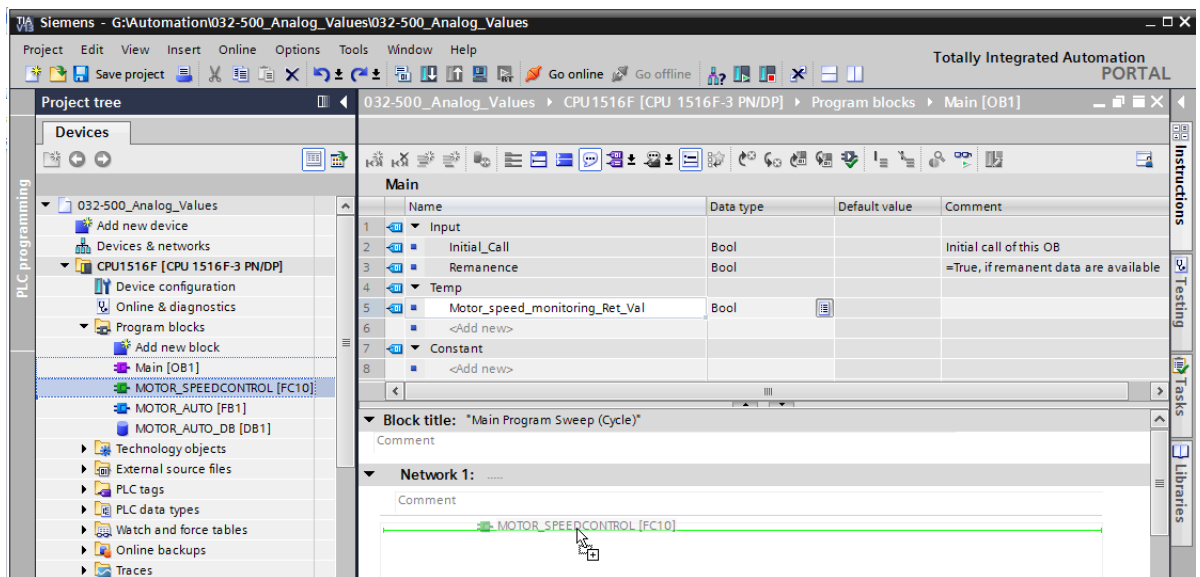


→ Select the block title of OB1 and then click  to insert a new Network 1 in front of the other networks

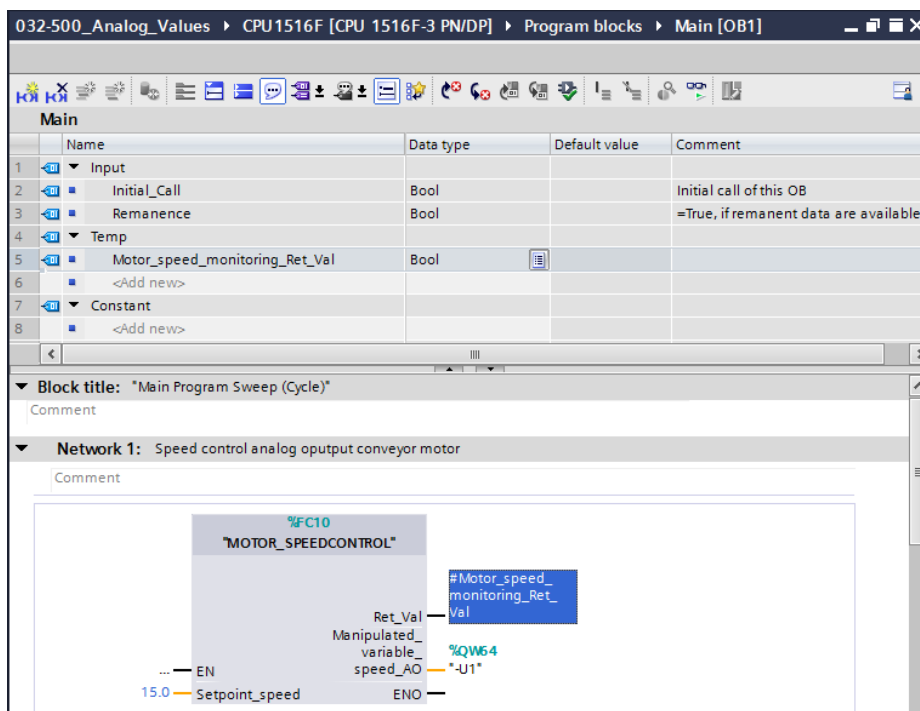
(→ )



→ Use drag-and-drop to move your "MOTOR_SPEEDCONTROL [FC10]" function onto the green line in Network 1.

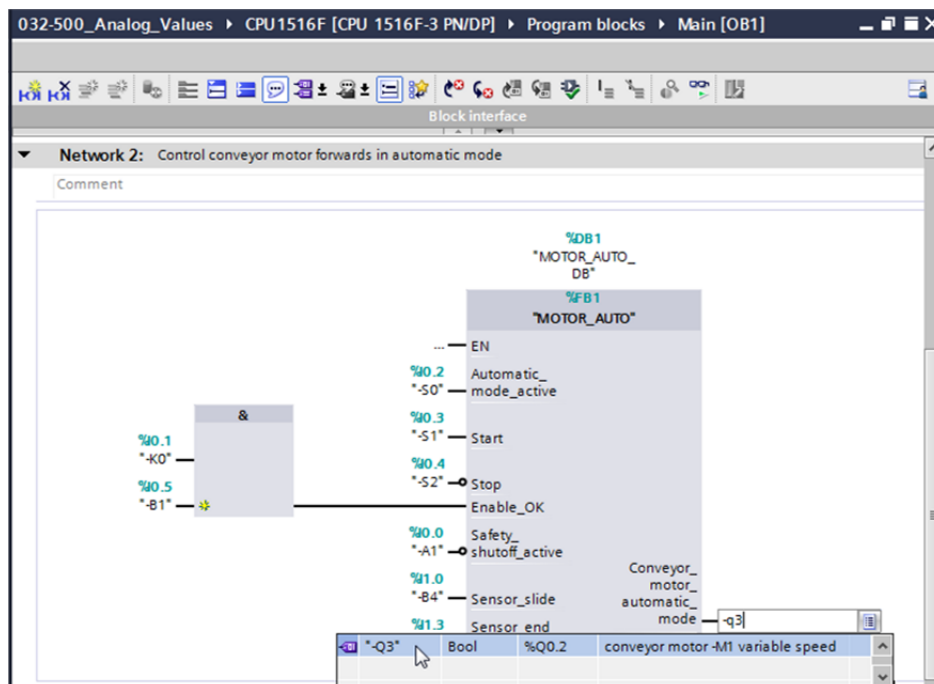


→ Connect the contacts with the constants and global and local tags here as shown.







→ Change the connection of output tag "Conveyor_motor_automatic_mode" in Network 2 to '-Q3' (Conveyor motor -M1 variable speed) so that the conveyor motor is controlled taking the analog speed setting into consideration.

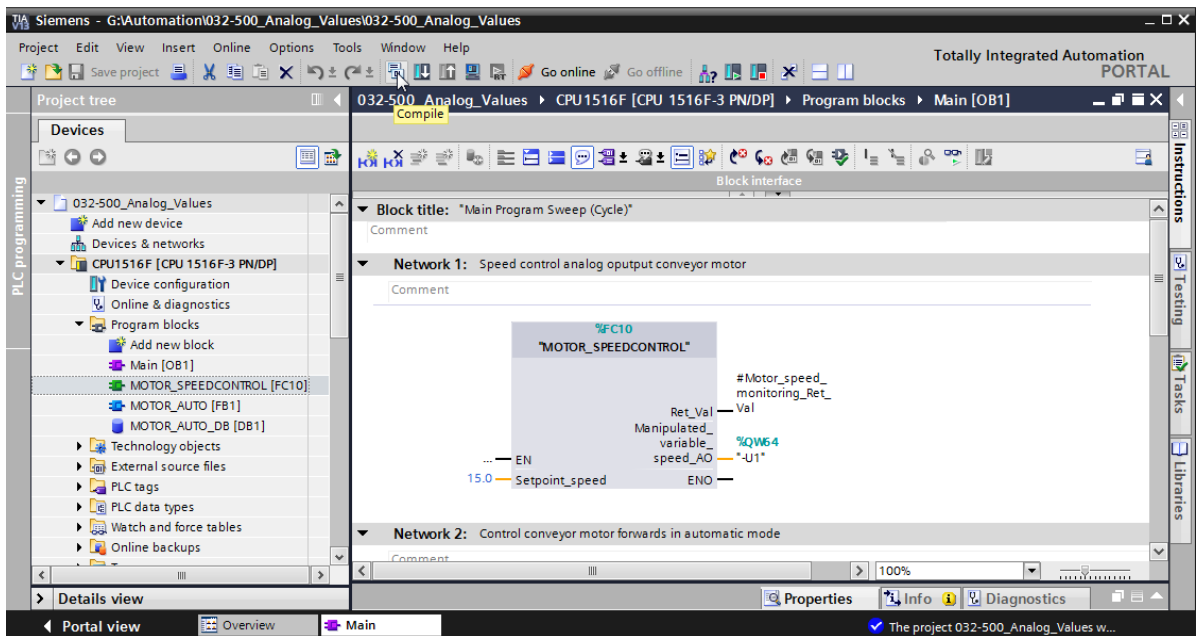
(→ -Q3)



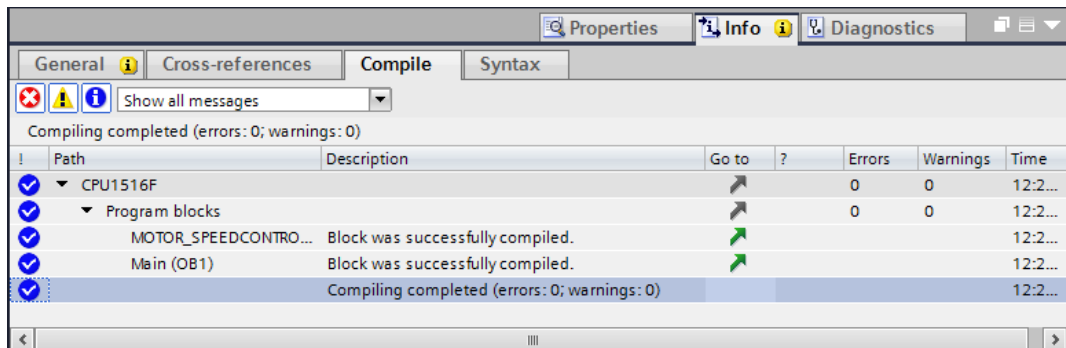
6.6 Save and compile the program

→ To save your project, select the  **Save project** button in the menu. To compile all blocks, click the "Program blocks" folder and select the  icon for compiling in the menu.


(→  **Save project** → Program blocks → )

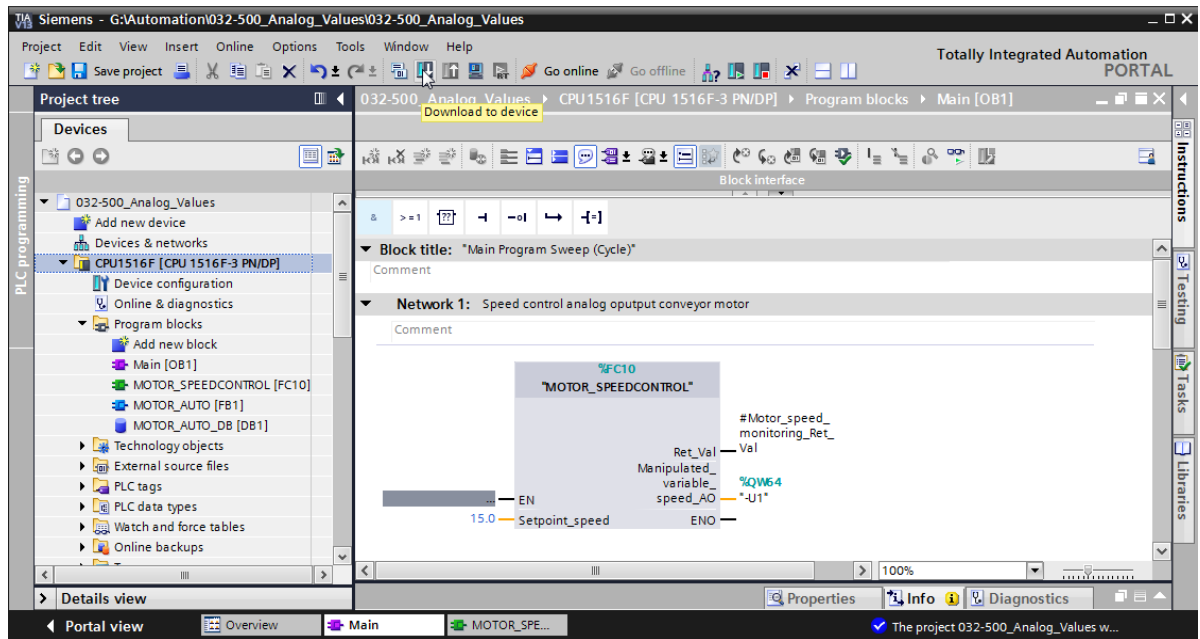


→ The "Info", "Compile" area shows which blocks were successfully compiled.





6.7 Download the program

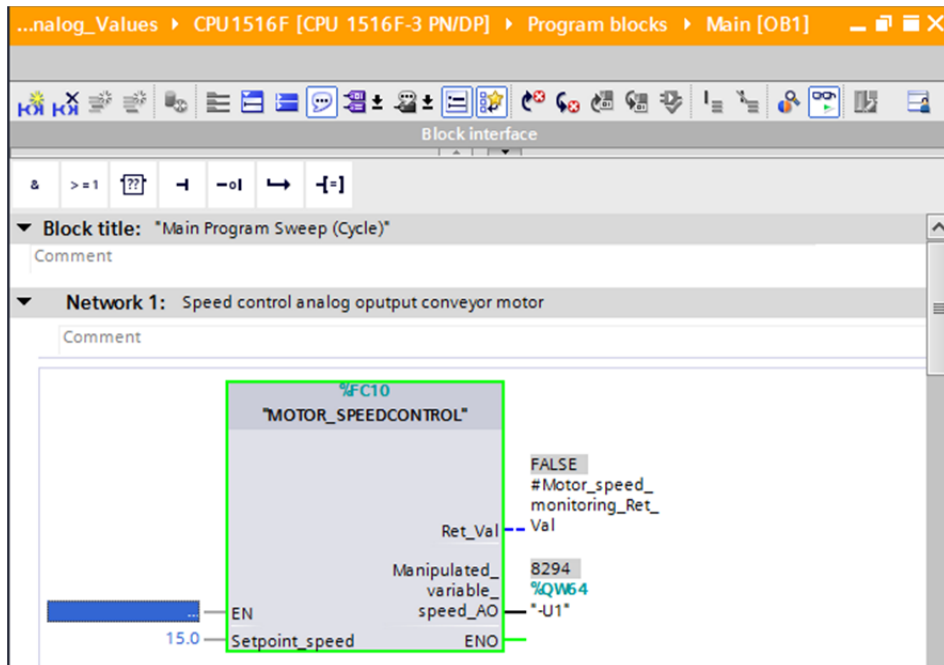
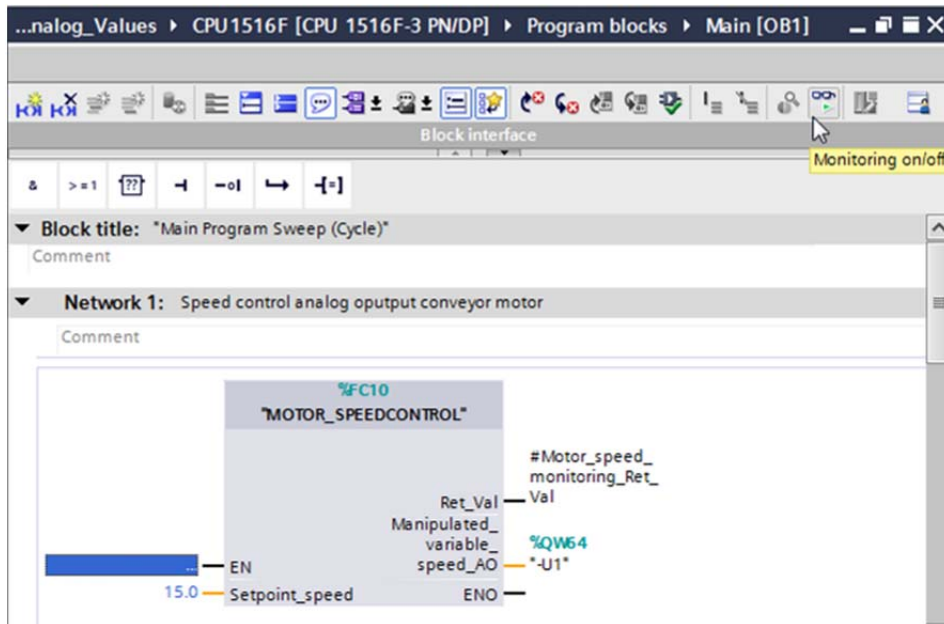
→ After successful compilation, the complete controller with the created program including the hardware configuration can, as described in the previous modules, be downloaded. (→ )



6.8 Monitor program blocks

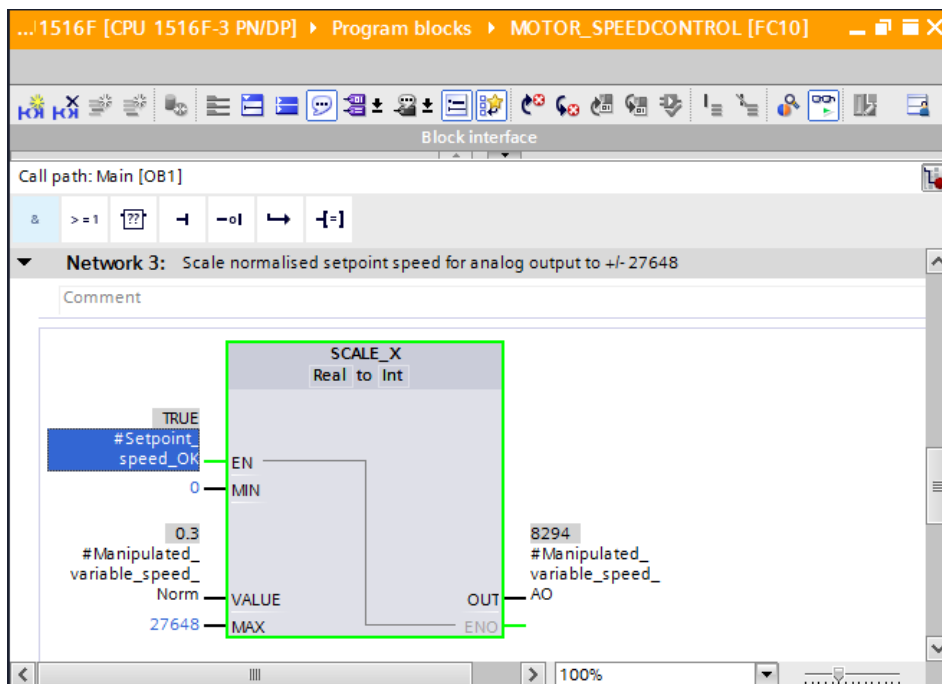
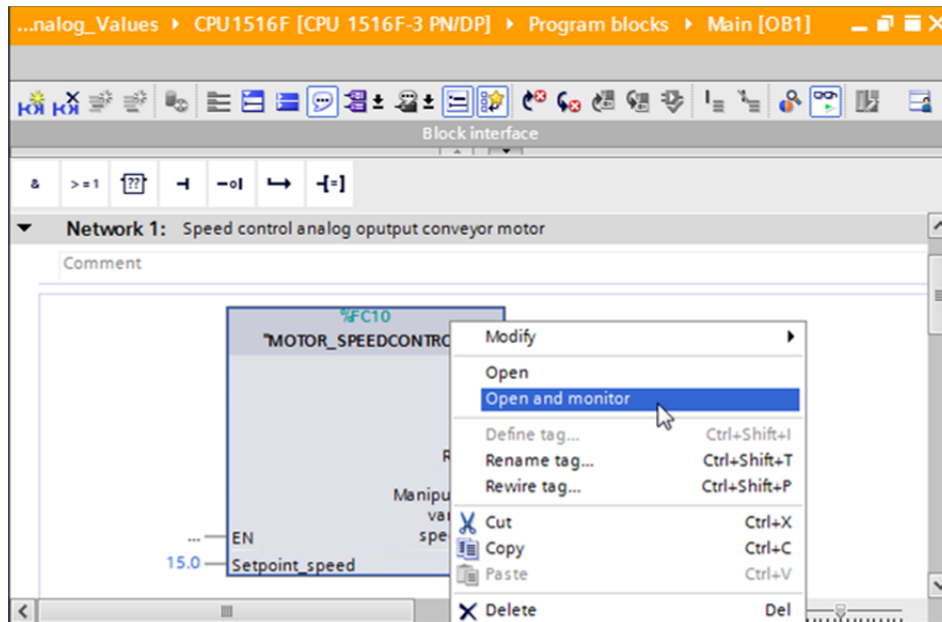
→ The desired block must be open for monitoring the downloaded program. The monitoring can now be activated/deactivated by clicking the  icon.

(→ Main [OB1] → )



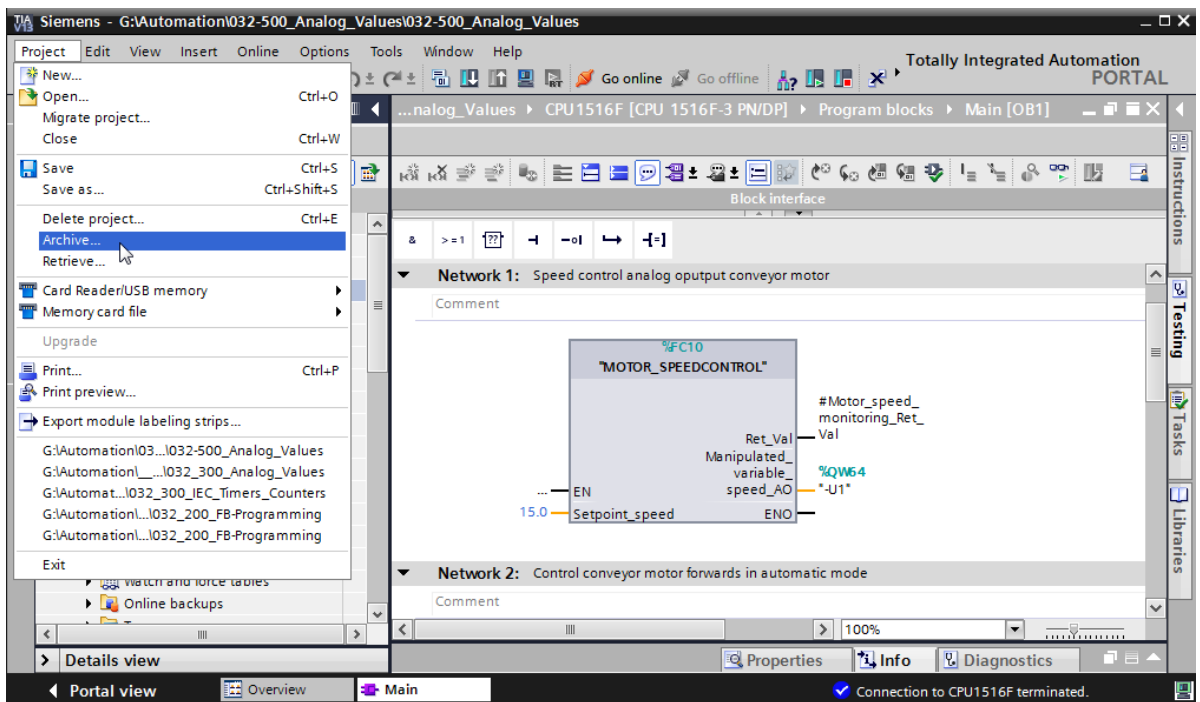
→ The "MOTOR_SPEEDCONTROL" [FC10] function called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking and the program code in the function can thus be monitored.

(→ "MOTOR_SPEEDCONTROL" [FC10] → Open and monitor)



6.9 Archive the project

- As the final step, we want to archive the complete project. Select the → 'Archive ...' command in the → 'Project' menu. Select a folder where you want to archive your project and save it with the file type "TIA Portal project archive".
- (→ Project → Archive → TIA Portal project archive → 032-500_Analog_Values.... → Save)



7 Checklist

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor part at slide activated (-B4 = 1) then Conveyor motor M1 variable speed (-Q3 = 1) switches on and stays on. The speed corresponds to the speed setpoint in the range +/- 50 rpm	
4	Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds)	
5	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0	
6	Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0	
7	Manual mode (-S0 = 0) → -Q3 = 0	
8	Switch off station (-K0 = 0) → -Q3 = 0	
9	Cylinder not retracted (-B1 = 0) → -Q3 = 0	
10	Project successfully archived	

8 Exercise

8.1 Task – Exercise

In this exercise a "MOTOR_SPEEDMONITORING" [FC11] function will be created additionally.

The actual value will be made available to -B8 (sensor actual value speed of the motor +/-10V corresponds to +/- 50 rpm) as an analog value and queried at an input of the "MOTOR_SPEEDMONITORING" [FC11] function. The data type is 16-bit integer (Int.).

This actual speed value will first be normalized to the range +/- 1 as 32-bit floating-point number (Real) in the function.

The normalized actual speed value will then be scaled to revolutions per minute (range: +/- 50 rpm) as 32-bit floating-point number (Real) and made available at an output.

The following 4 limit values can be specified as 32-bit floating-point numbers (Real) at the block inputs in order to monitor them in the function:

Speed > Motor_speed_monitoring_error_max

Speed > Motor_speed_monitoring_warning_max

Speed < Motor_speed_monitoring_warning_min

Speed < Motor_speed_monitoring_error_min

If a limit value is exceeded or fallen below, the value TRUE (1) is assigned to the corresponding output bit.

If a fault is present, the protective tripping of the "MOTOR_AUTO" [FB1] function block will be tripped.

8.2 Technology diagram

Here you see the technology diagram for the task.

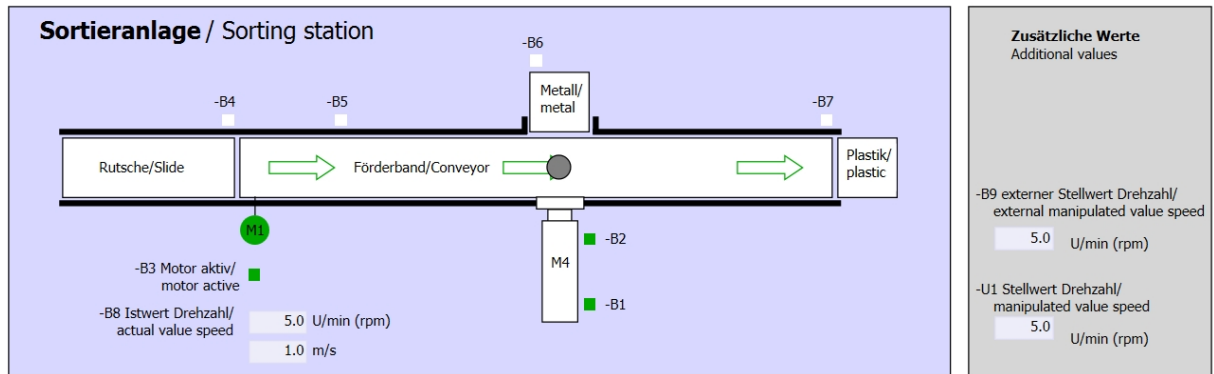


Figure 3: Technology diagram

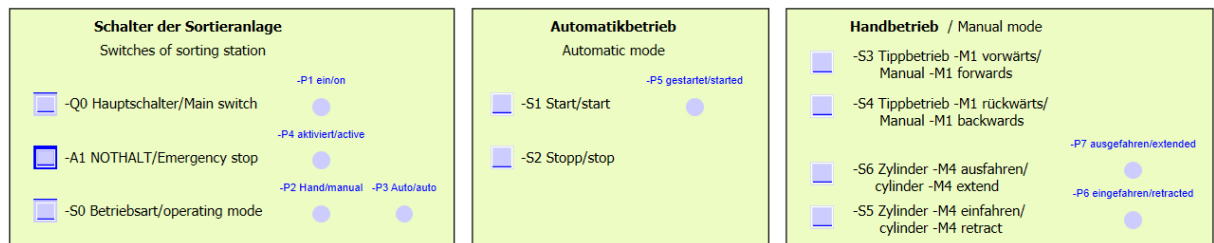


Figure 4: Control panel

8.3 Reference list

The following signals are required as global operands for this task.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop OK	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	-S1	Pushbutton automatic start	NO
I 0.4	BOOL	-S2	Pushbutton automatic stop	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO
I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO
IW64	BOOL	-B8	Sensor actual value speed of the motor +/-10V corresponds to +/- 50 rpm	

DO	Type	Identifier	Function	
Q 0.2	BOOL	-Q3	Conveyor motor -M1 variable speed	
QW 64	BOOL	-U1	Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm	

Legend for reference list

DI Digital Input

DO Digital Output

AI Analog Input

AO Analog Output

I Input

Q Output

NC Normally Closed

NO Normally Open

8.4 Planning

Plan the implementation of the task on your own.

8.5 Checklist – Exercise

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor part at slide activated (-B4 = 1) then Conveyor motor -M1 variable speed (-Q3 = 1) switches on and stays on. The speed corresponds to the speed setpoint in the range +/- 50 rpm	
4	Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds)	
5	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0	
6	Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0	
7	Manual mode (-S0 = 0) → -Q3 = 0	
8	Switch off station (-K0 = 0) → -Q3 = 0	
9	Cylinder not retracted (-B1 = 0) → -Q3 = 0	
10	Speed > Motor_speed_monitoring_error_max → -Q3 = 0	
11	Speed < Motor_speed_monitoring_error_min → -Q3 = 0	
12	Project successfully archived	

9 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Please note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, please contact your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, especially Prof. Dr.-Ing. Leon Urbas und Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	4
2	Prerequisite.....	4
3	Theory.....	4
3.1	Data blocks	4
3.2	Data types of the SIMATIC S7-1500	5
3.3	Optimized blocks	6
3.4	Download without reinitialization	6
4	Task	7
5	Planning.....	7
5.1	Global data block for speed control and speed monitoring of the motor.....	7
5.2	Technology diagram	8
5.3	Reference list.....	9
6	Structured step-by-step instructions	10
6.1	Retrieve an existing project	10
6.2	Create the global data block "SPEED_MOTOR"	12
6.3	Access to data of the data block in the organization block.....	17
6.4	Save and compile the program.....	21
6.5	Download the program	22
6.6	Monitor/modify values in data blocks.....	23
6.7	Initialize setpoints/reset start values.....	24
6.8	Snapshots in data blocks.....	26
6.9	Expand data block and download it without reinitialization	30
6.10	Archive the project.....	34
7	Checklist	35
8	Exercise	36
8.1	Task – Exercise	36
8.2	Technology diagram	36
8.3	Reference list.....	37
8.4	Planning.....	37
8.5	Checklist – Exercise	38
9	Additional information	39

GLOBAL DATA BLOCKS FOR THE SIMATIC S7-1500

1 Goal

In this chapter, you will become acquainted with the use of global data blocks for the SIMATIC S7-1500 with the TIA Portal programming tool.

The module explains the structure and creation of and access to global data blocks for the SIMATIC S7-1500. It also shows the steps for creating a global data block in the TIA Portal and for accessing this data in the program with read and write access.

2 Prerequisite

This chapter builds on the chapter Analog Values with the SIMATIC S7 CPU1516F-3 PN/DP. You can use the following project for this chapter, for example: "SCE_EN_032-500_Analog_Values_R1508.zap13".

3 Theory

3.1 Data blocks

In contrast to logic blocks, data blocks contain no instructions. Rather, they serve as memory for user data.

Data blocks thus contain variable data that is used by the user program. You can define the structure of global data blocks as required.

Global data blocks store data that can be used **by all other blocks** (see Figure 1). Only the associated function block should access instance data blocks. The maximum size of data blocks varies depending on the utilized CPU.

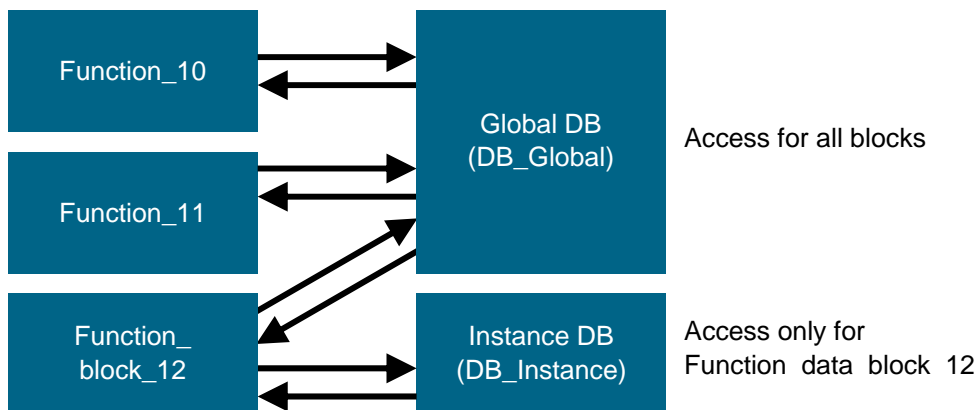


Figure 1: Difference between global DB and instance DB.

Application examples for **global data blocks** are:

- Saving of information about a storage system. "Which product is located where?"
- Saving of recipes for particular products.

The data in data blocks is stored retentively in most cases. This data is then retained in the event of a power failure or after a STOP/START of the CPU.

3.2 Data types of the SIMATIC S7-1500

The SIMATIC S7-1500 has many different data types for representing different numerical formats. A list of some of the elementary data types is given below.

Data type	Size (bits)	Range	Example of constant entry
Bool	1	0 to 1	TRUE, FALSE, 0, 1
Byte	8	16#00 to 16#FF	16#12, 16#AB
Word	16	16#0000 to 16#FFFF	16#ABCD, 16#0001
DWord	32	16#00000000 to 16#FFFFFFFF	16#02468ACE
Char	8	16#00 to 16#FF	'A', 'r', '@'
Sint	8	-128 to 127	123, -123
Int	16	-32,768 to 32,767	123, -123
Dint	32	-2,147,483,648 to 2,147,483,647	123, -123
USInt	8	0 to 255	123
UInt	16	0 to 65,535	123
UDInt	32	0 to 4,294,967,295	123
Real	32	$\pm 1.18 \times 10^{-38}$ to $\pm 3.40 \times 10^{38}$	123.456, -3.4, 1.2E+12 3.4E-3
LReal	64	$\pm 2.23 \times 10^{-308}$ to $\pm 1.79 \times 10^{308}$	12345.123456789 -1.2E+40
Time	32	T#-24d_20h_31 m_23s_648ms to T#24d_20h_31 m_23s_647ms Saved as: -2,147,483,648 ms to +2,147,483,647 ms	T#5m_30s 5#-2d T#1d_2h_15m_30x_45ms
String	Variable	0 to 254 characters in byte size	'ABC'
Array		With arrays, data of a uniform data type is arranged one after the other and addressed consecutively in the address area. The properties of each array element are identical and are configured in the array tag.	
Struct		The STRUCT data type represents a data structure that consists of a fixed number of components of different data types. Components of STRUCT or ARRAY data type can also be nested in a structure.	
...		For other data types, refer to the online help.	

3.3 Optimized blocks

S7-1500 controllers have optimized data storage. In optimized blocks all tags are automatically sorted based on their data type. The sorting ensures that data gaps between the tags are minimized and the tags are stored in a manner that optimizes their access by the controller.

- The tags are always accessed as fast as possible because the file storage by the system is optimized and is independent of the declaration.
- There is no danger of inconsistencies due to incorrect, absolute accesses because symbolic access is generally used.
- Declaration changes do not result in access errors because accesses by process visualization systems, for example, occur symbolically.
- Individual tags can be selectively defined as retentive.
- No settings are needed or possible in the instance data block. Everything will be set in the assigned FB (e.g., retentivity).
- Memory reserves in the data block enable changes to be made without loss of actual values (download without reinitialization).

3.4 Download without reinitialization

To enable the subsequent editing of user programs that are already running in a CPU, the S7-1500 controllers support the option of expanding the interfaces of optimized function or data blocks during operation. You can download the modified blocks without switching the controller to STOP mode and without affecting the actual values of previously downloaded tags.

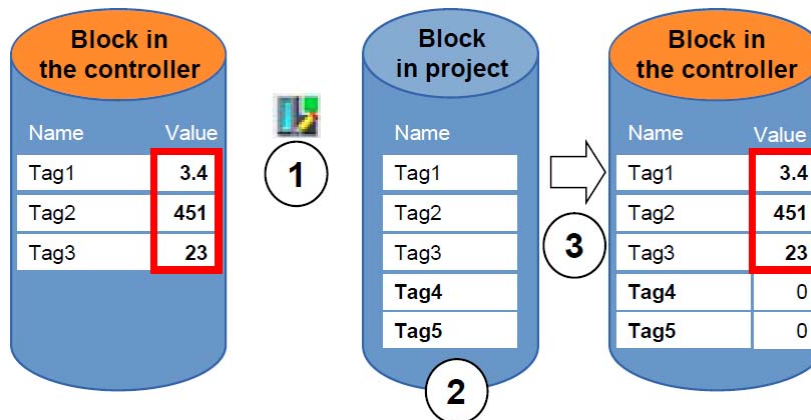


Figure 2: Download without reinitialization

The following steps can be performed while the controller is in RUN mode:

1. Activate "Download without reinitialization"
2. Insert newly defined tags in an existing block
3. Download expanded block to the controller

The newly defined tags are initialized. The existing tags retain their current value.

Prerequisite: a memory reserve must have been defined for the block beforehand and the block with this memory reserve must have downloaded to the CPU.

4 Task

In this chapter, the program from chapter "SCE_EN_032-500 Analog Values" will be expanded to include a data block that centrally provides the parameters for the two functions "MOTOR_SPEEDCONTROL" [FC10] and "MOTOR_SPEEDMONITORING" [FC11].

5 Planning

The data management and setpoint setting for the "MOTOR_SPEEDCONTROL" [FC10] and "MOTOR_SPEEDMONITORING" [FC11] functions will be carried out using the global data block "SPEED_MOTOR" [DB2].

This will be added to the "032-500_Analog_Values" project. This project must be retrieved from the archive beforehand.

In the "Main" [OB1] organization block, the two functions "MOTOR_SPEEDCONTROL" [FC10] and "MOTOR_SPEEDMONITORING" [FC11] must then be connected with the tags from global data block "SPEED_MOTOR" [DB2].

5.1 Global data block for speed control and speed monitoring of the motor

Speed setpoint and actual speed value will be created in Real data format (32-bit floating-point number) as the first tags in the "SPEED_MOTOR" [DB2] data block. The speed setpoint is thereby given the start value + 14 U/min

A structure (Struct) 'Positive_Speed' will then be created for monitoring the positive speed limits.

This structure contains the 2 tags 'Threshold_Error' (start value + 15 rpm) and 'Threshold_Warning' (start value + 10 rpm) in Real data format (32-bit floating-point number) and the 2 tags 'Error' and 'Warning' in Bool data format (binary number).

The structure (Struct) 'Positive_Speed' will then be inserted again as a copy and renamed to 'Negative_Speed' for monitoring the negative speed limits.

The 'Threshold_Error' tag is given the start value - 16 rpm and the 'Threshold_Warning' tag the start value - 14 rpm.

5.2 Technology diagram

Here you see the technology diagram for the task.

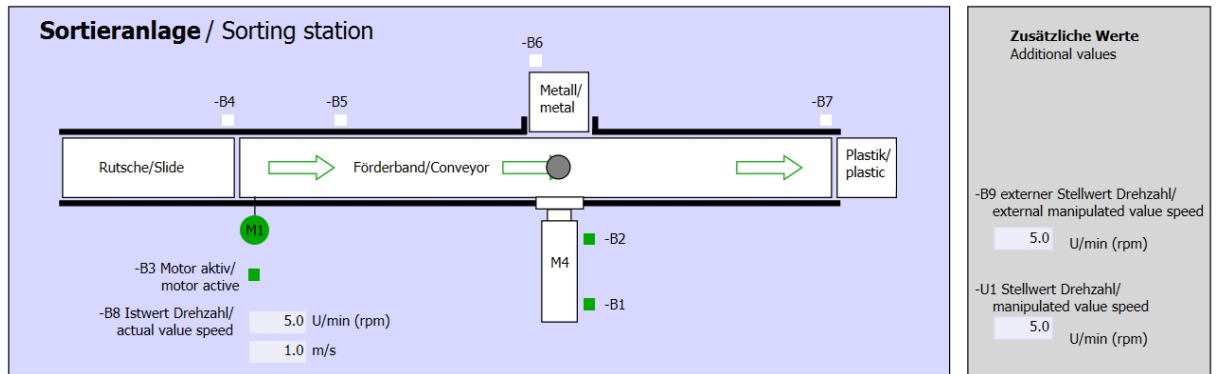


Figure 3: Technology diagram

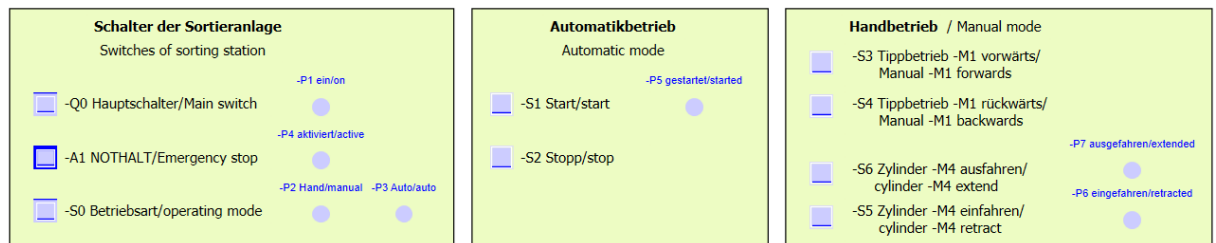


Figure 4: Control panel

5.3 Reference list

The following signals are required as global operands for this task.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop OK	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	-S1	Pushbutton automatic start	NO
I 0.4	BOOL	-S2	Pushbutton automatic stop	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO
I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO
IW64	BOOL	-B8	Sensor actual value speed +/-10V corresponds to +/- 50 rpm	

DO	Type	Identifier	Function	
Q 0.2	BOOL	-Q3	Conveyor motor -M1 variable speed	
QW 64	BOOL	-U1	Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm	

Legend for reference list

DI Digital Input

DO Digital Output

AI Analog Input

AO Analog Output

I Input

Q Output

NC Normally Closed

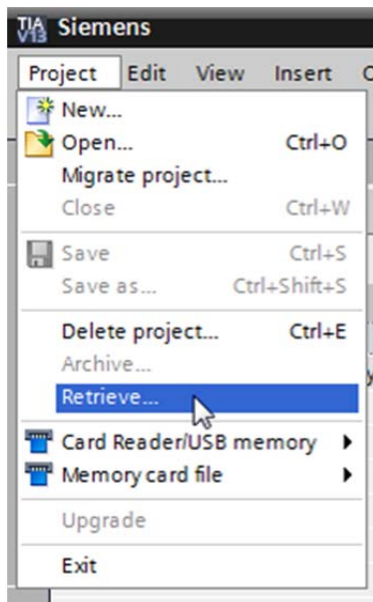
NO Normally Open

6 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

6.1 Retrieve an existing project

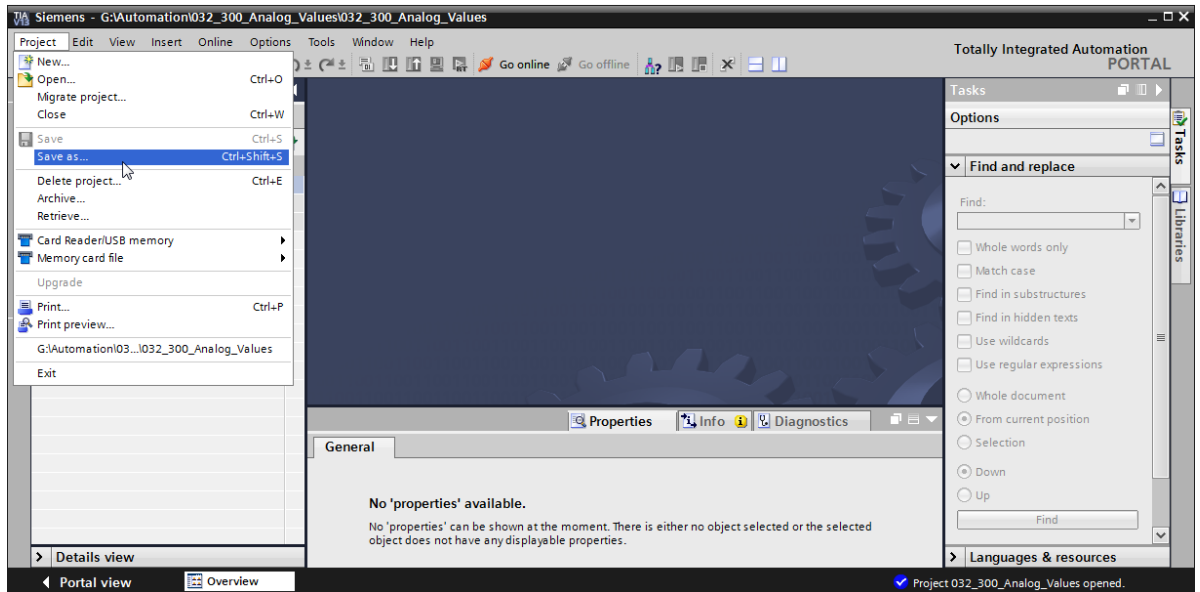
- Before we can expand the "SCE_EN_032-500_Analog_Values_R1508.zap13" project from chapter "SCE_EN_032-500 Analog Values", we must retrieve this project from the archive. To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open.
(→ Project → Retrieve → Select a .zap archive → Open)



- The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK".
(→ Target directory → OK)

→ Save the opened project under the name 032-600_Global_Data_Blocks.

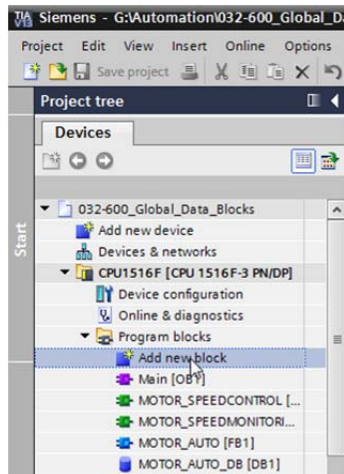
(→ Project → Save as ... → 032-600_Global_Data_Blocks → Save)




6.2 Create the global data block "SPEED_MOTOR"

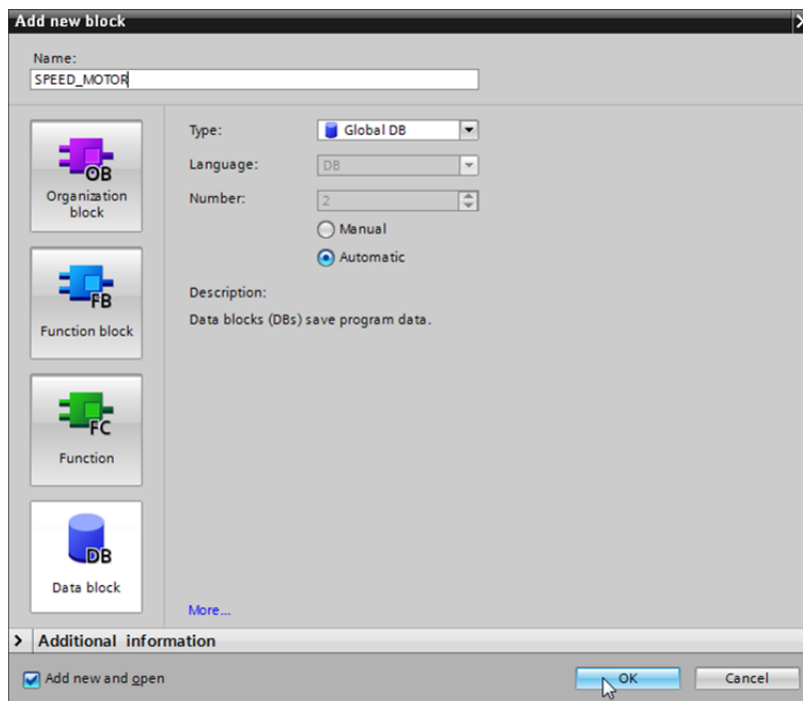
→ Select the 'Program blocks' folder of your CPU 1516F-3 PN/DP and then click "Add new block" to create a new global data block there.

(→ CPU_1516F [CPU 1516F-3 PN/DP] → Add new block)



→ Select  in the next dialog and rename your new block to: "SPEED_MOTOR". Select 'Global DB' as the type. The number '2' will be automatically assigned. Select the "Add new and open" check box. Click "OK".

(→  → Name: SPEED_MOTOR → Type: Global DB → ☒ Add new and open → OK)



→ The "SPEED_MOTOR" data block is automatically displayed. Start by creating the 'Speed_Setpoint' and 'Speed_Actual_Value' tags shown here with their associated comments. Select 'Real' as the data type. Also set a start value of 10.0 rpm for the 'Speed_Setpoint'.

(→ Speed_Setpoint → Real → 14.0 → Speed_Actual_Value → Real)

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Static			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)

Note: Be sure to use the correct data types.

→ Next we create a tag structure 'Struct' so it can be duplicated later.

(→ Struct)

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Static			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	<Add new>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

→ Name the structure 'Positive_Speed' and enter a comment.

(→ Positive_Speed)

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR								
	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	▼ Static							
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	▼ Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning positive speed
5	<Add new>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
6	<Add new>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

→ Create the tags for the speed monitoring below the structure as shown here.

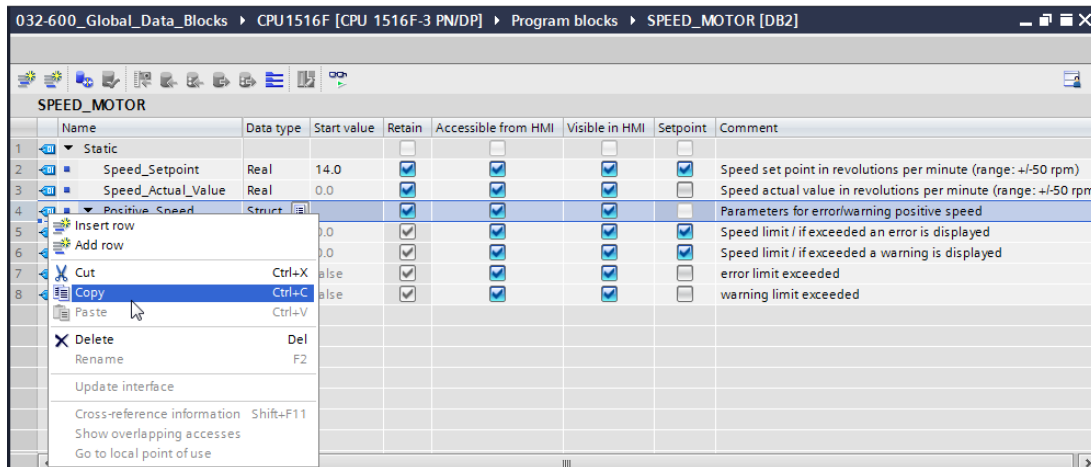
032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR								
	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	▼ Static							
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	▼ Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded

Note: Be sure to use the correct data types.

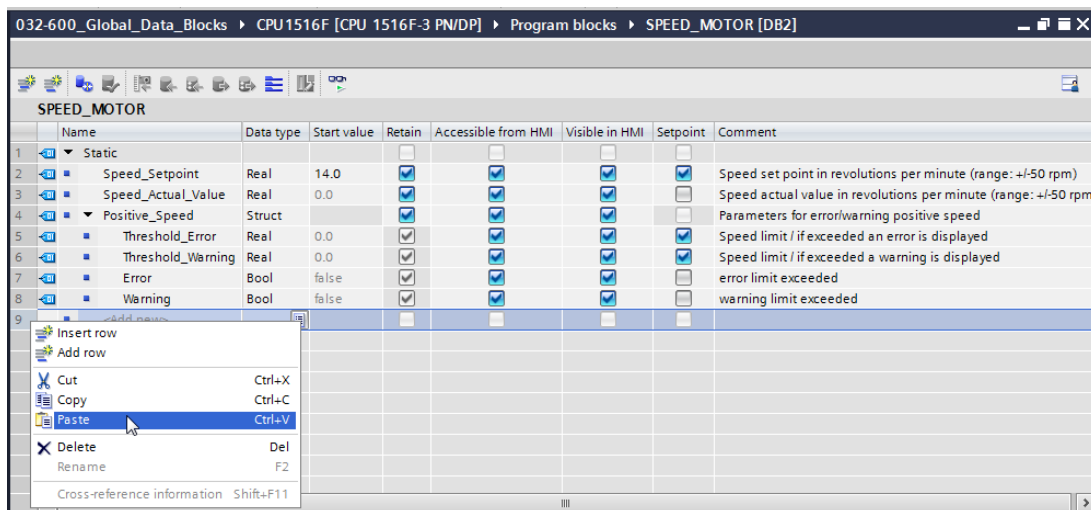
→ Then select the structure and copy it.

(→ Copy)



→ Paste the copied structure below the 'Positive_Speed' structure again.

(→ Paste)



- Rename the new structure to 'Negative_Speed' and enter a comment.
(→ Negative_Speed)

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Static							
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning negative speed
10	Threshold_Error	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
14	<Add new>							

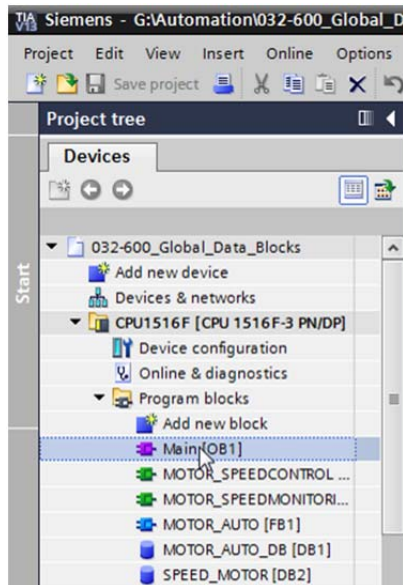
Do not forget to click Save project. The finished global data block "SPEED_MOTOR" [DB2] is shown below. Check to verify that Retain is selected ☒ and the corresponding start value is entered for all tags. The data will thus be retained in the data block even after a power failure or a STOP/START of the CPU. The check boxes for 'Accessible from HMI' ☒ and 'Visible in HMI' ☒ should also all have a check mark so that all tags in future expansions of this project will be accessible by the visualization systems (HMI). We will select the 'Setpoint' check box ☒ only for the default values in our data block.
(→ ☒ ☒ ☒ ☒)

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Static							
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning negative speed
10	Threshold_Error	Real	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
14	<Add new>							

Note: The use of setpoints is described further below in the step-by-step instructions of the module.

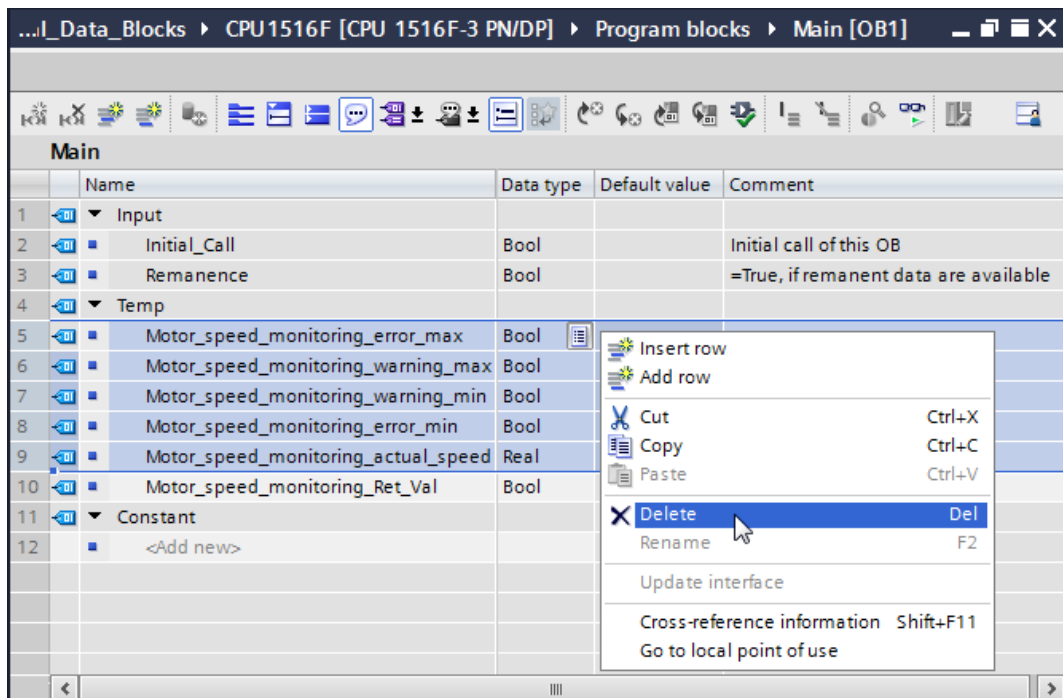
6.3 Access to data of the data block in the organization block



→ Open the "Main [OB1]" organization block with a double-click.

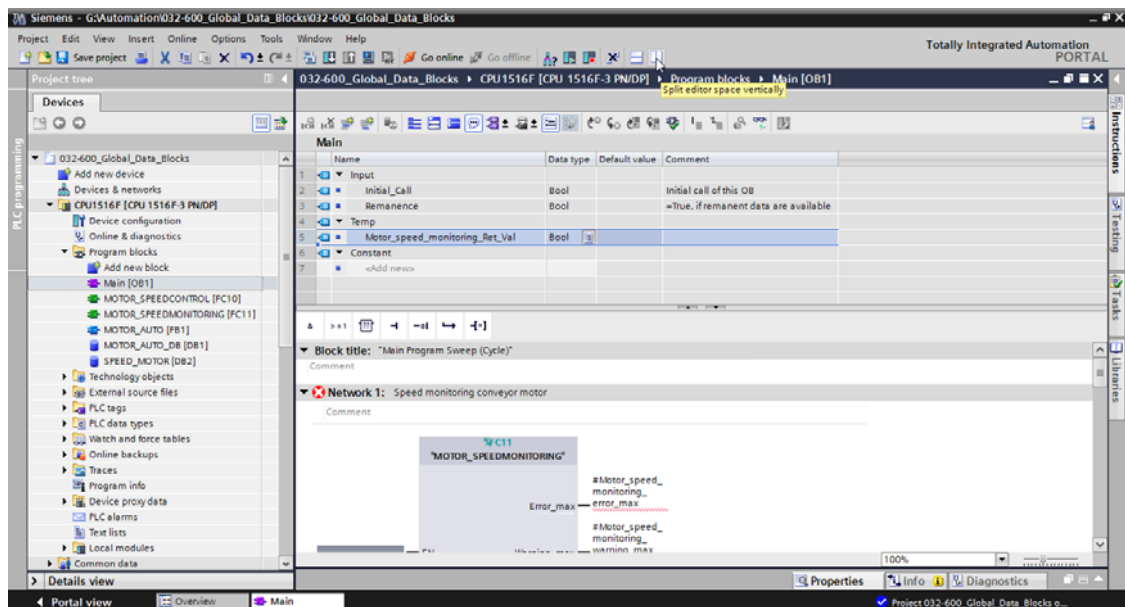


→ Delete the temporary tags in Main [OB1] that are no longer needed. Only the Boolean tag 'Motor_Speed_Control_Ret_Val' is still needed.

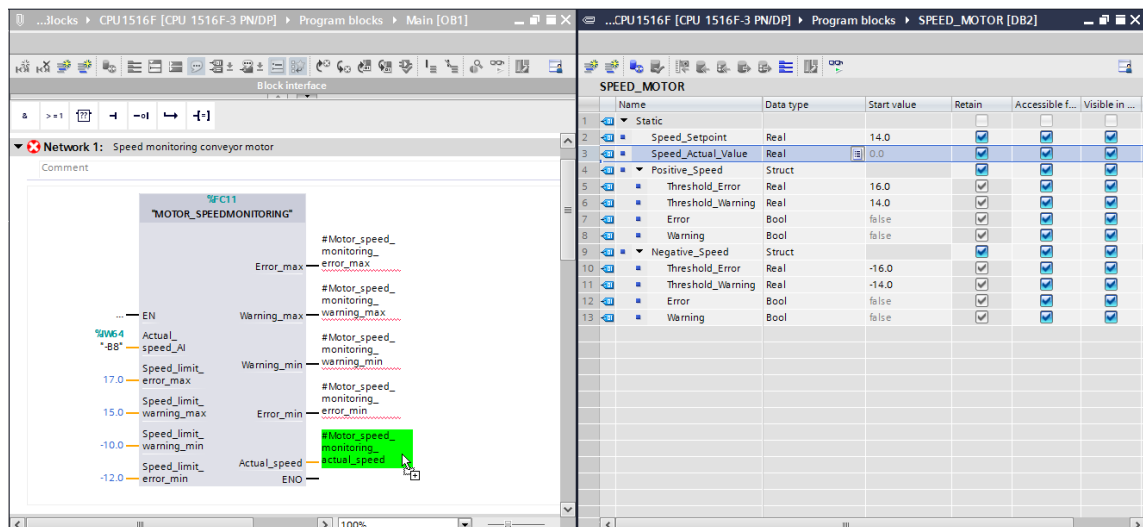
(→ Delete)



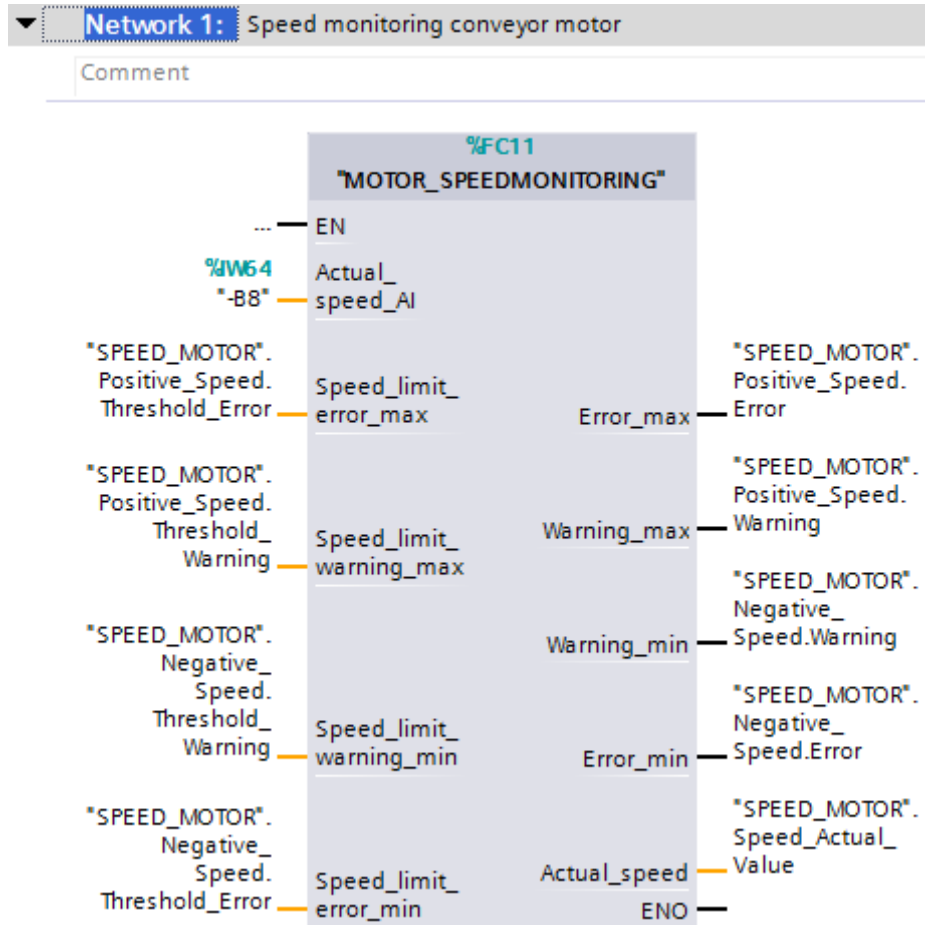
- Have the "SPEED_MOTOR" [DB2] data block and the "Main" [OB1] organization block displayed side by side by clicking the  icon to vertically split the editor area.
- (→ )



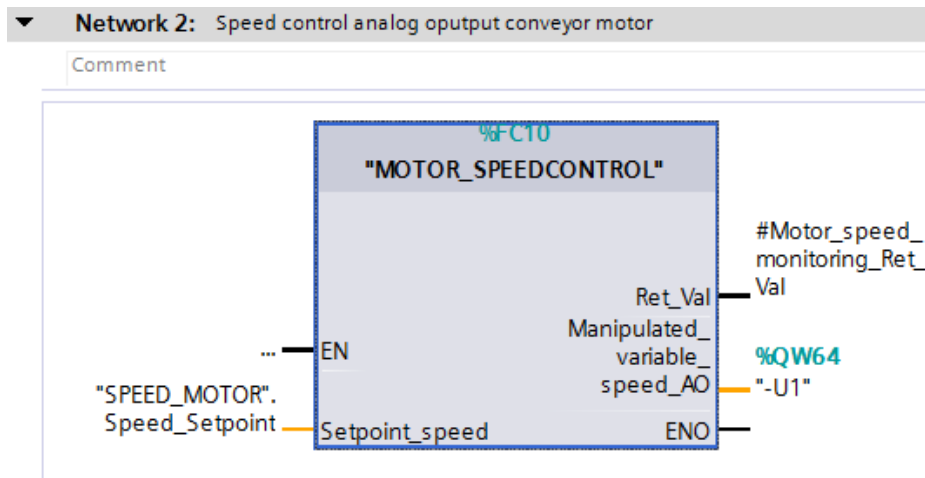
- Use drag-and-drop to move the tags needed for the interconnection from the "SPEED_MOTOR" [DB2] data block onto the connections of the called functions and function blocks in the "Main" [OB1] organization block. First we move the 'Speed_Actual_Value' tag onto the 'Actual_speed' output of the "MOTOR_SPEEDMONITORING" [FC11] block.
- (→ Speed_Actual_Value)



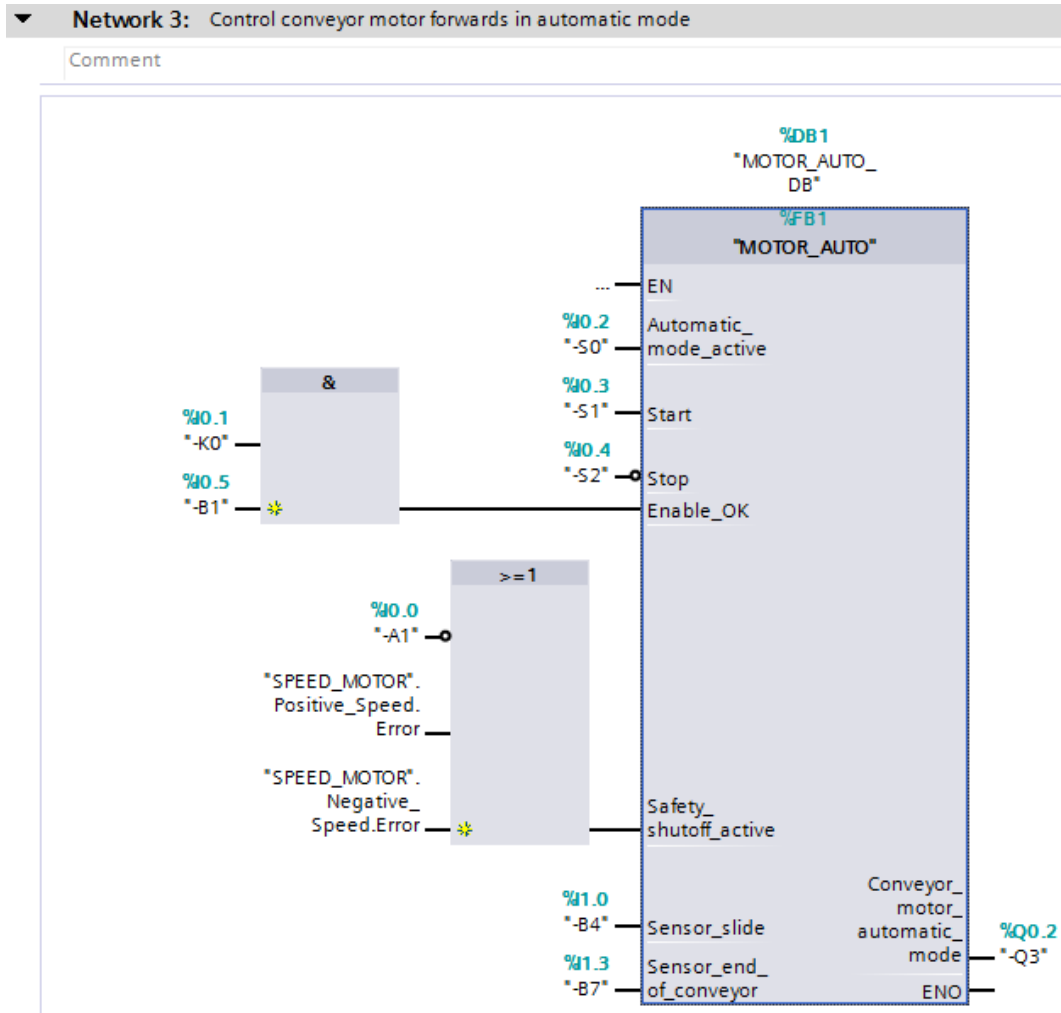
- Also connect the other contacts in Network 1 with tags from the "SPEED_MOTOR" [DB2] data block as shown here.





- Connect the contacts in Network 2 with tags from the "SPEED_MOTOR" [DB2] data block as shown here.





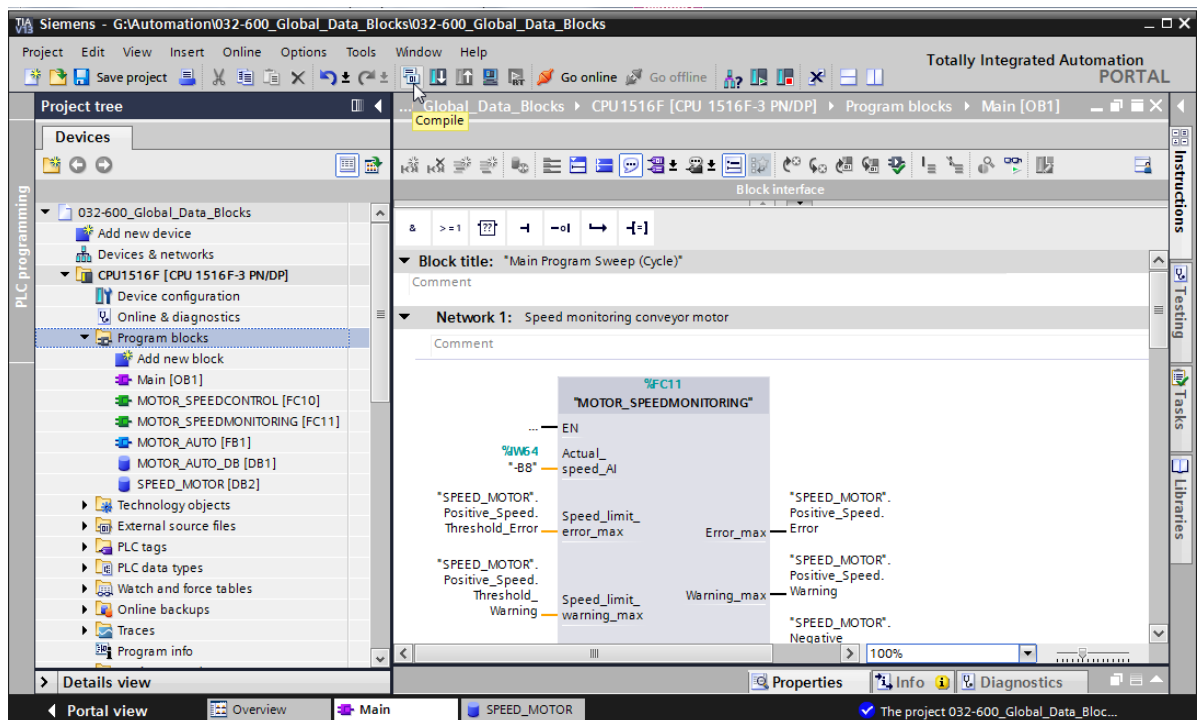
→ Connect the contacts in Network 3 with tags from the "SPEED_MOTOR" [DB2] data block as shown here.



6.4 Save and compile the program

→ To save your project, click the  **Save project** button in the menu. To compile all blocks, click the "Program blocks" folder and select the  icon for compiling in the menu.


(→  **Save project** → Program blocks → )

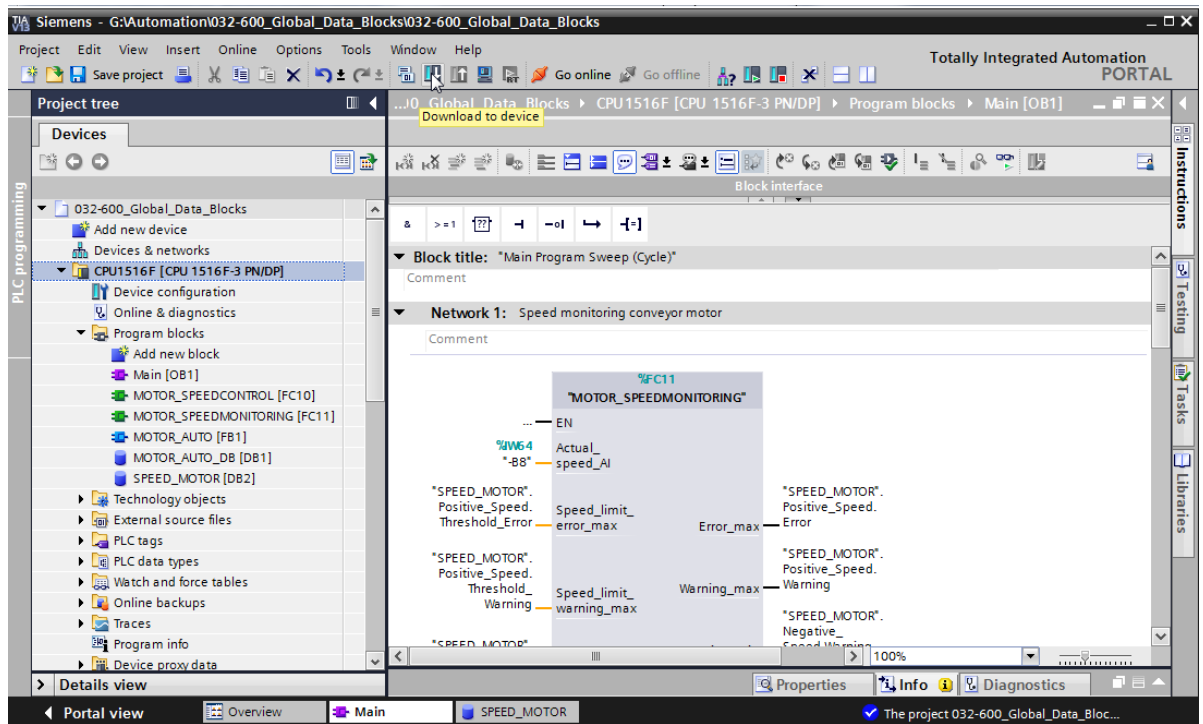


→ The "Info", "Compile" area shows which blocks were successfully compiled.


Properties Info Diagnostics							
General Cross-references Compile Syntax							
Show all messages							
Compiling completed (errors: 0; warnings: 0)							
!	Path	Description	Go to	?	Errors	Warnings	Time
✓	▼ CPU1516F		↗		0	0	7:26:42 AM
✓	▼ Program blocks		↗		0	0	7:26:42 AM
✓	SPEED_MOTOR (DB2)	Block was successfully compiled.	↗				7:26:42 AM
✓	Main (OB1)	Block was successfully compiled.	↗				7:26:44 AM
✓	Compiling completed (errors: 0; warnings: 0)						7:26:48 AM

6.5 Download the program

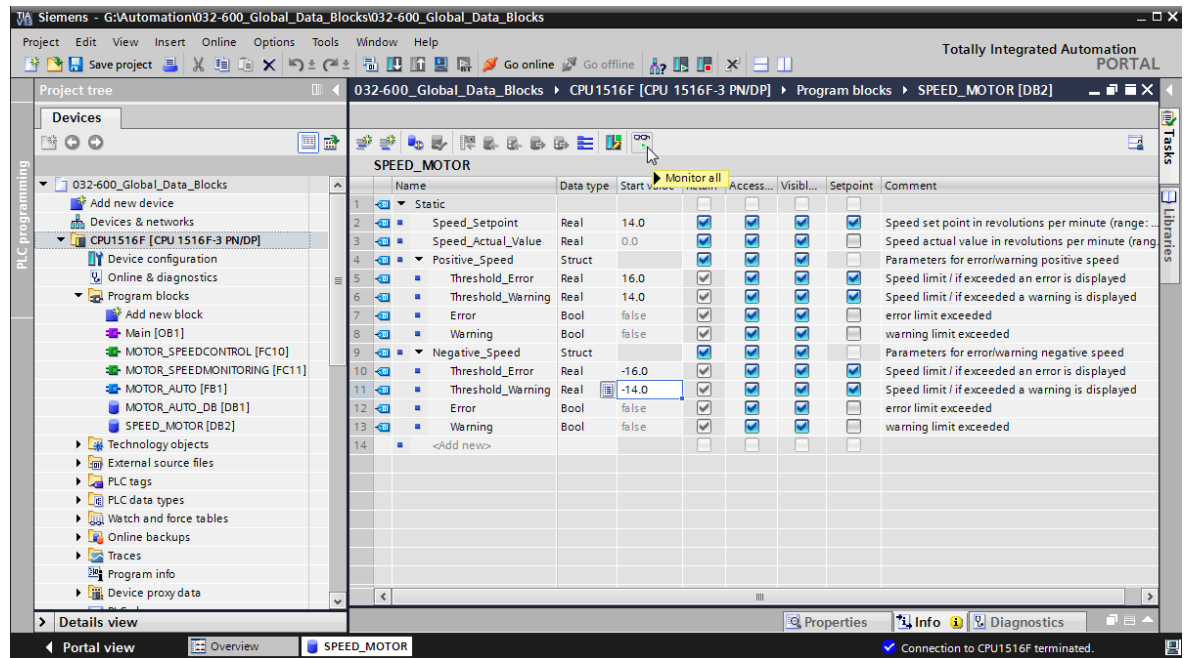
→ After successful compilation, the complete controller with the created program including the hardware configuration can, as described in the previous modules, be downloaded. (→ )



6.6 Monitor/modify values in data blocks

→ The desired block must be open for monitoring the tags of a downloaded data block. The monitoring can then be activated/deactivated by clicking the  icon.

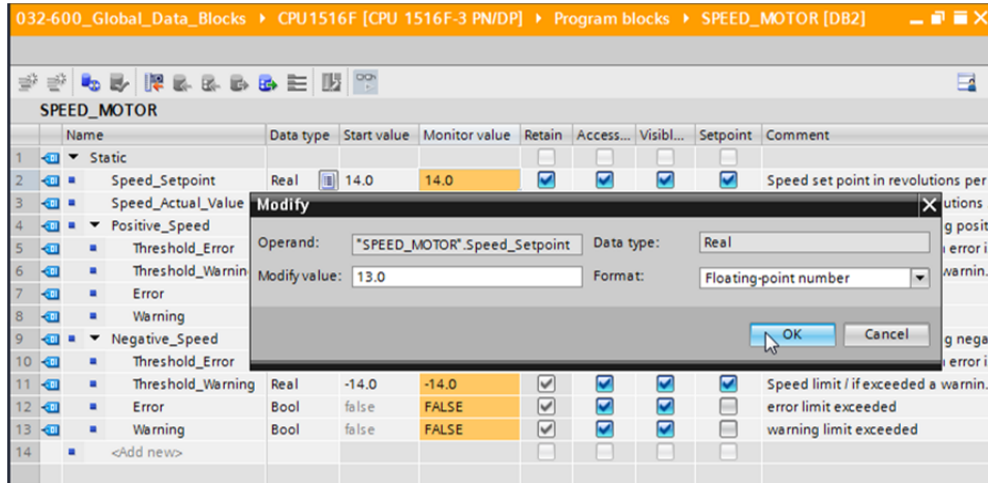
(→ SPEED_MOTOR [DB2] → )



→ In the 'Monitor value' column, the values currently available in the CPU can be monitored.

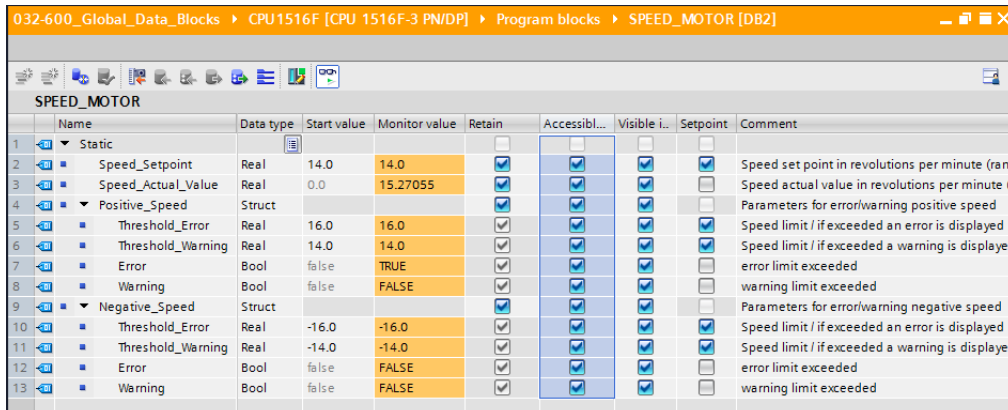
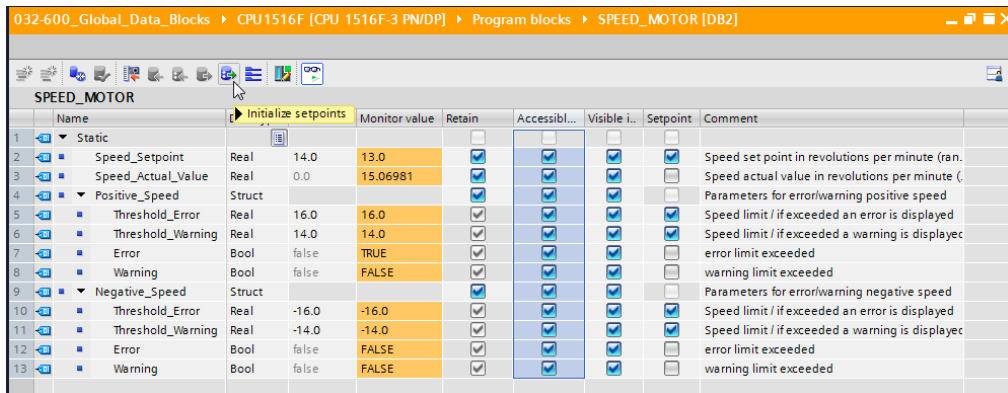
032-600_Global_Data_Blocks > CPU1516F [CPU 1516F-3 PN/DP] > Program blocks > SPEED_MOTOR [DB2]									
SPEED_MOTOR									
	Name	Data type	Start value	Monitor value	Retain	Access...	Visibl...	Setpoint	Comment
1	Static				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2	Speed_Setpoint	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per .
3	Speed_Actual_Value	Real	0.0	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions ...
4	Positive_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning posit...
5	Threshold_Error	Real	16.0	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error i...
6	Threshold_Warning	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warnin...
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning nega...
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error i...
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warnin...
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
14	<Add new>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	


- If you right-click on one of the values, the 'Modify' dialog for modifying this value opens (→ Modify → Modify value: 14.0 → OK)



6.7 Initialize setpoints/reset start values


- The setpoints can be initialized by clicking the 'Initialize setpoints' icon. For the tags whose 'Setpoint' check box is selected ☒, the start value will then be applied as the current value.



→ All start values can be reset by clicking the  icon.


(→ )

032-600_Global_Data_Blocks ▶ CPU 1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR  Reset start values


	Name	Data type	Start value	Monitor value	Retain	Accessibl...	Visible i...	Setpoint	Comment
1	Static								
2	Speed_Setpoint	Real	14.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran.
3	Speed_Actual_Value	Real	0.0	15.27055	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (.
4	Positive_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	16.0	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displaye
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning negative speed
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displaye
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded

032-600_Global_Data_Blocks ▶ CPU 1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR 

	Name	Data type	Start value	Monitor value	Retain	Accessibl...	Visible i...	Setpoint	Comment
1	Static								
2	Speed_Setpoint	Real	0.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran.
3	Speed_Actual_Value	Real	0.0	15.27055	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (.
4	Positive_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	0.0	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	0.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displaye
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning negative speed
10	Threshold_Error	Real	0.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	0.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displaye
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded

6.8 Snapshots in data blocks

→ If you click the  icon, a snapshot of the monitored values can be taken in order to apply these values as start values or to transfer them back to the CPU later

(→ ).

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR (snapshot created: 7/29/2015 7:53:09 AM)



Snapshot of the monitored values

	Name		Start value	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	Static								
2	Speed_Setpoint	Real	14.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolu..
3	Speed_Actual_Value	Real	0.0	15.19097	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in rev.
4	Positive_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/war...
5	Threshold_Error	Real	16.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded...
6	Threshold_Warning	Real	14.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded...
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/war...
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded...
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded...
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)

	Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1	Static									
2	Speed_Setpoint	Real	14.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3	Speed_Actual_Value	Real	0.0	15.19097	15.19097	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Spee...
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
5	Threshold_Error	Real	16.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6	Threshold_Warning	Real	14.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...

→ Alternatively, values from the snapshot can be applied by clicking the  icon for all values or by clicking the  icon for the start values only. Only the setpoints are needed here in most cases

(→ )

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)

▶ Copy all values from the "Snapshot" column to the "Start value" column

Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1 Static									
2 Speed_Setpoint	Real	14.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3 Speed_Actual_Value	Real	0.0	15.19097	15.64308	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Spee...
4 Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
5 Threshold_Error	Real	16.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6 Threshold_Warning	Real	14.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7 Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
8 Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...
9 Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
10 Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11 Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12 Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
13 Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)

▶ Copy all setpoints from the "Snapshot" column to the "Start value" column


Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1 Static									
2 Speed_Setpoint	Real	14.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3 Speed_Actual_Value	Real	0.0	15.19097	15.64308	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Spee...
4 Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
5 Threshold_Error	Real	16.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6 Threshold_Warning	Real	14.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7 Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
8 Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...
9 Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
10 Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11 Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12 Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
13 Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...


032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)

Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1 Static									
2 Speed_Setpoint	Real	13.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3 Speed_Actual_Value	Real	0.0	15.19097	15.64308	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Spee...
4 Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
5 Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6 Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7 Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
8 Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...
9 Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
10 Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11 Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12 Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
13 Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...

SCE Training Curriculum | TIA Portal Module 032-600, Edition 02/2016 | Digital Factory, DF FA

→ To transfer data captured in the snapshot back to the CPU, you must click '.

(→ )

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)


Copy all values from the snapshot to the actual values of the CPU

	Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1	Static									
2	Speed_Setpoint	Real	13.0	13.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3	Speed_Actual_Value	Real	0.0	15.19097	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
5	Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6	Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warni...
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warni...

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)

	Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1	Static									
2	Speed_Setpoint	Real	13.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3	Speed_Actual_Value	Real	0.0	15.19097	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
5	Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6	Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warni...
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warni...

→ If you want to overwrite all setpoints with the start values, you can initiate this by clicking ''. The values in the CPU for which the 'Setpoint' check box was not selected are thereby retained.

(→ )

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)

Initialize setpoints

	Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1	Static									
2	Speed_Setpoint	Real	13.0	13.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3	Speed_Actual_Value	Real	0.0	15.19097	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Spee...
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
5	Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6	Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

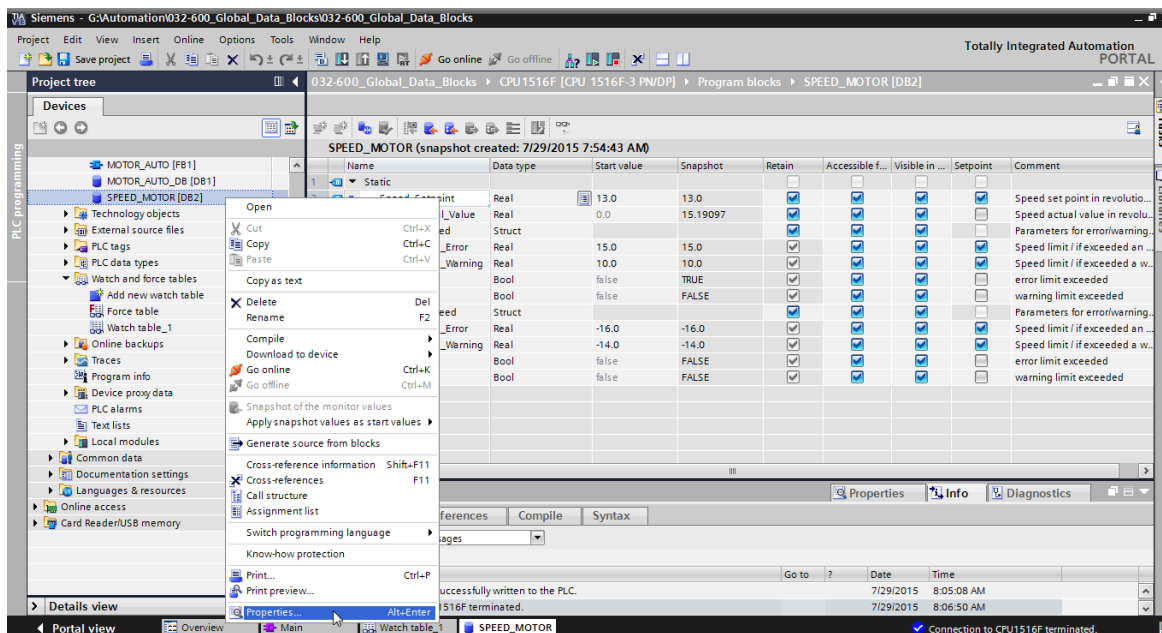
SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)

	Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1	Static									
2	Speed_Setpoint	Real	13.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3	Speed_Actual_Value	Real	0.0	15.19097	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Spee...
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
5	Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6	Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Para...
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error ...
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warni...

6.9 Expand data block and download it without reinitialization

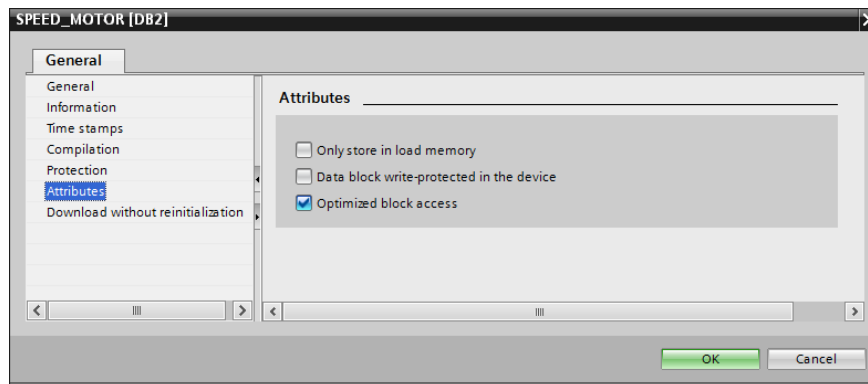
→ To enable 'Download without reinitialization' for the "SPEED_MOTOR" [DB2] data block, you must 'Go offline' and then open the properties of the data block.

(→ Go offline → SPEED_MOTOR [DB2] → Properties)

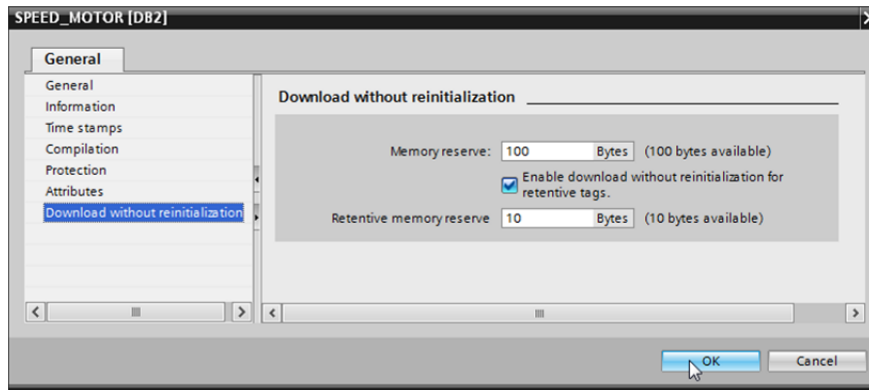


→ Select the 'Optimized block access' check box ☒ in the properties under 'General', 'Attributes'.

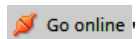
(→ General → Attributes → ☒ Optimized block access)



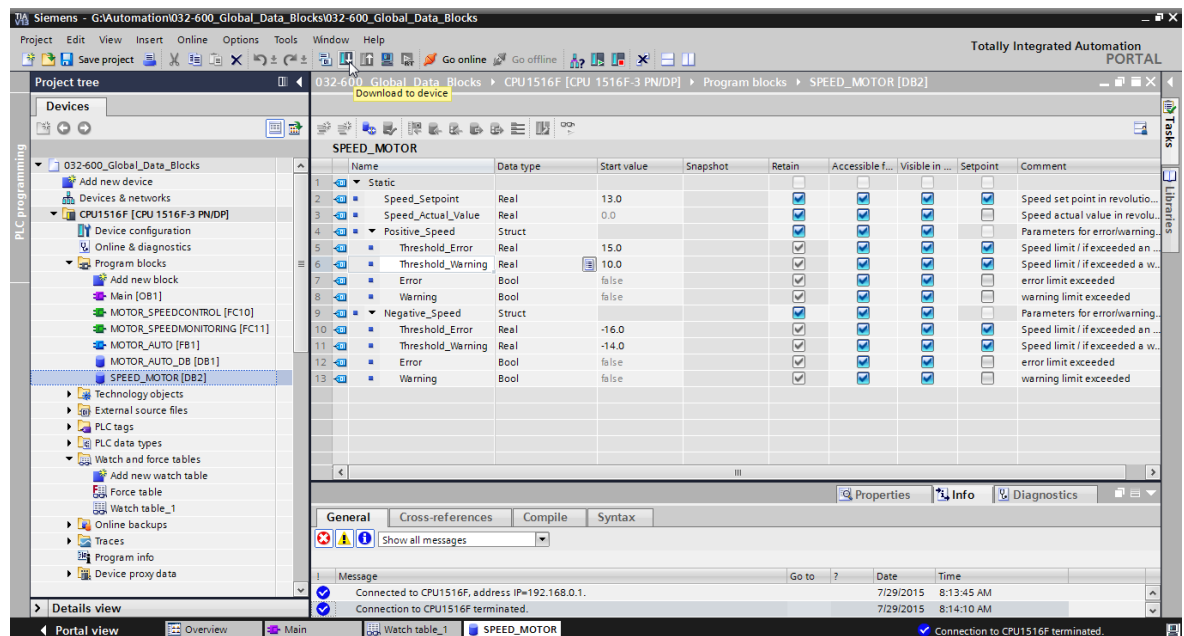
- Assign a 'Retentive memory reserve' to the data block for 'Download without reinitialization'.
 (→Download without reinitialization → Retentive memory reserve → 10 bytes → OK)




- Download your "SPEED_MOTOR" [DB] data block to the controller again and select 'Go online'.



(→SPEED_MOTOR [DB] →  → )



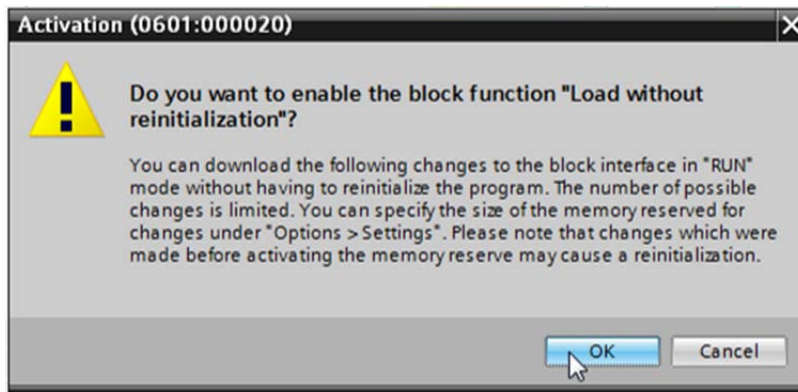
→ Then click the  icon to activate download without reinitialization and confirm the safety prompt with 'OK'.

(→  → OK)

032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR


	Name	Data type	Download without reinitialization	Accessibl...	Visible i...	Setpoint	Comment
1	▼ Static						
2	Speed_Setpoint	Real	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran.
3	Speed_Actual_Value	Real	0.0	15.27055	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (.
4	▼ Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9	▼ Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning negative speed
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded



- Next add any tag in your data block
 (→ Name: Value_test → Data type: Real → Start value: 99)

032-600_Global_Data_Blocks > CPU1516F [CPU 1516F-3 PN/DP] > Program blocks > SPEED_MOTOR [DB2]

	Name	Data type	Start value	Retain	Accessibl...	Visible i...	Setpoint	Comment
1	Static							
2	Speed_Setpoint	Real	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning negative speed
10	Threshold_Error	Real	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
14	Value_Test	Real	99.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
15	<Add new>							

- Download your "SPEED_MOTOR" [DB] data block to the controller again.
 (→SPEED_MOTOR [DB] →  → Download)

Siemens - G\Automation\032-600_Global_Data_Blocks\032-600_Global_Data_Blocks

Project Edit View Insert Online Options Tools Window Help

Save project Go online Go offline

Project tree

- 032-600_Global_Data_Blocks
 - Devices & networks
 - CPU1516F [CPU 1516F-3 PN/DP]
 - Device configuration
 - Online & diagnostics
 - Program blocks
 - SPEED_MOTOR [DB2]
 - Download to device

SPEED_MOTOR

	Name	Data type	Start value	Retain	Accessibl...	Visible i...	Setpoint	Comment
1	Static							
2	Speed_Setpoint	Real	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parameters for error/warning negative speed
10	Threshold_Error	Real	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	warning limit exceeded
14	Value_Test	Real	99.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
15	<Add new>							

General Cross-references Compile Syntax

1 Message

	Message	Go to	Date	Time
✓	5 setpoint values successfully written to the PLC.		7/30/2015	6:29:17 PM
✓	Connection to CPU1516F terminated.		7/30/2015	6:30:59 PM
✓	Connected to CPU1516F, address IP=192.168.0.1		7/30/2015	6:32:49 PM

Portal view Overview Watch table, 1 SPEED_MOTOR

Connected to CPU1516F, address IP=1...


Load preview

Check before loading

Status	I	Target	Message	Action
✓	<input checked="" type="checkbox"/>	CPU1516F	Ready for loading.	
✓	<input checked="" type="checkbox"/>	Software	Download software to device	Consistent download
✓	<input checked="" type="checkbox"/>	Overwrite online?	Objects that exist online and are overwritten.	
✓	<input checked="" type="checkbox"/>	Main [OB1]		<input checked="" type="checkbox"/> Overwrite
✓	<input checked="" type="checkbox"/>	SPEED_MOTOR [DB2]		<input checked="" type="checkbox"/> Overwrite

Refresh

Finish Load Cancel

→ If you click  to monitor the block again, you will see that the monitored values have not been overwritten with the start values.

(→ )

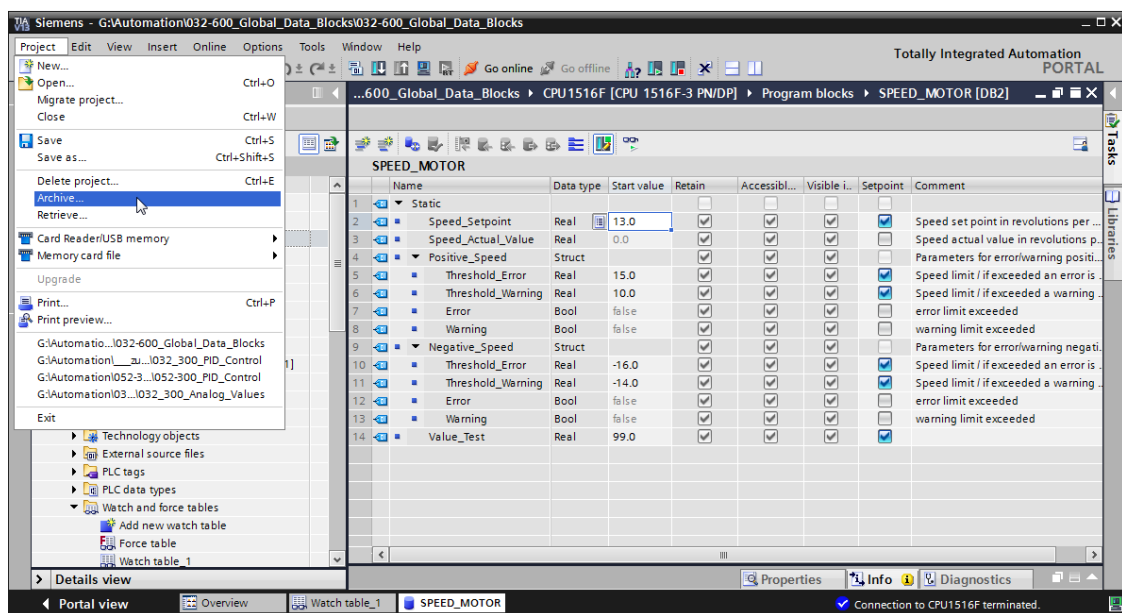
032-600_Global_Data_Blocks ▶ CPU1516F [CPU 1516F-3 PN/DP] ▶ Program blocks ▶ SPEED_MOTOR [DB2]

SPEED_MOTOR									
	Name	Data type	Start value	Monitor value	Retain	Accessibl...	Visible i...	Setpoint	Comment
1	Static								
2	Speed_Setpoint	Real	13.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran...
3	Speed_Actual_Value	Real	0.0	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (...)
4	Positive_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	15.0	17.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	10.0	12.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning negative speed
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
14	Value_Test	Real	99.0	99.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

6.10 Archive the project

→ As the final step, we want to archive the complete project. Select the → 'Archive ...' command in the → 'Project' menu. Select a folder where you want to archive your project and save it with the file type "TIA Portal project archive".

(→ Project → Archive → TIA Portal project archive → 032-600_Global_Data_Blocks.... → Save)



7 Checklist

No.	Description	Completed
1	Data block SPEED_MOTOR [DB2] successfully created.	
2	Program changes made in Main [OB1].	
3	Compiling successful and without error message	
4	Download successful and without error message	
5	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor part at slide activated (-B4 = 1) then Conveyor motor M1 variable speed (-Q3 = 1) switches on and stays on. The speed corresponds to the speed setpoint in the range +/- 50 rpm	
6	Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds)	
7	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0	
8	Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0	
9	Manual mode (-S0 = 0) → -Q3 = 0	
10	Switch off station (-K0 = 0) → -Q3 = 0	
11	Cylinder not retracted (-B1 = 0) → -Q3 = 0	
12	Speed > Motor_speed_monitoring_error_max → -Q3 = 0	
13	Speed < Motor_speed_monitoring_error_min → -Q3 = 0	
14	Project successfully archived	

8 Exercise

8.1 Task – Exercise

In this exercise a global data block "MAGAZINE_PLASTIC" [DB3] will be created additionally.

The setpoint and actual value of the counter for the plastic parts will be specified and displayed in this data block.

A connectable input for the setpoint setting and an output for displaying the actual value will also be added to the "MOTOR_AUTO" [FB1] function block.

8.2 Technology diagram

Here you see the technology diagram for the task.

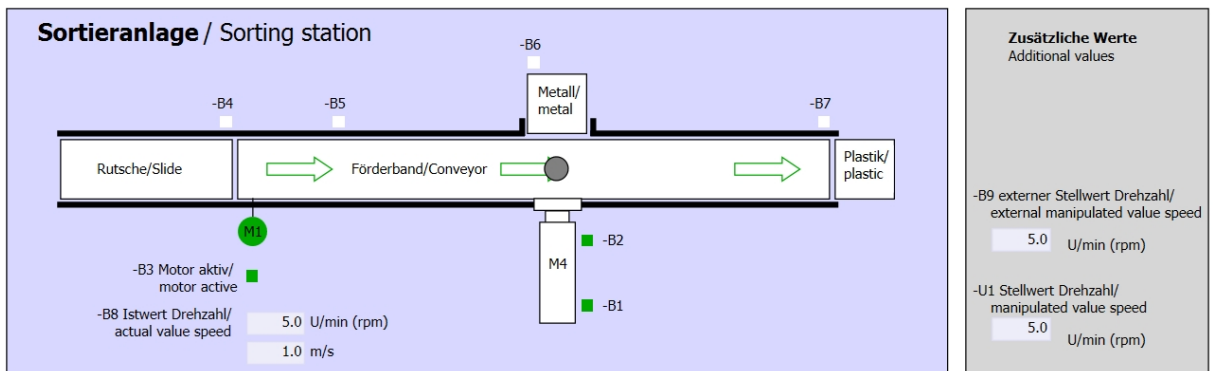


Figure 5: Technology diagram

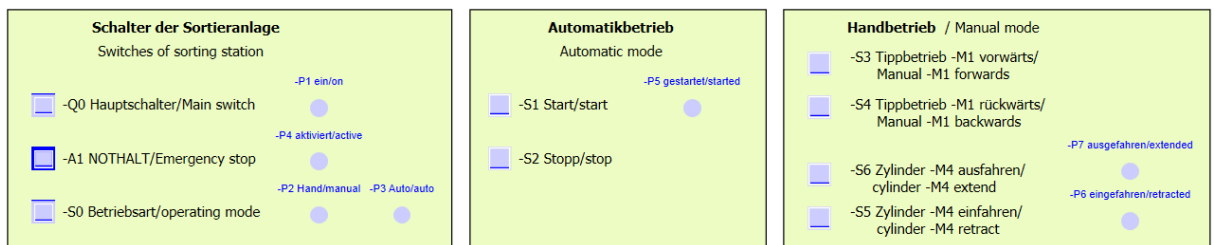


Figure 6: Control panel

8.3 Reference list

The following signals are required as global operands for this task.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop OK	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	-S1	Pushbutton automatic start	NO
I 0.4	BOOL	-S2	Pushbutton automatic stop	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO
I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO
IW64	BOOL	-B8	Sensor actual value speed of the motor +/-10V corresponds to +/- 50 U/min	

DO	Type	Identifier	Function	
Q 0.2	BOOL	-Q3	Conveyor motor -M1 variable speed	
QW 64	BOOL	-U1	Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm	

Legend for reference list

DI Digital Input

DO Digital Output

AI Analog Input

AO Analog Output

I Input

Q Output

NC Normally Closed

NO Normally Open

8.4 Planning

Plan the implementation of the task on your own.

8.5 Checklist – Exercise

No.	Description	Completed
1	Data block MAGAZINE_PLASTIC [DB3] successfully created.	
2	Program changes made in MOTOR_AUTO [FB1].	
3	Program changes made in Main [OB1].	
4	Compiling successful and without error message	
5	Download successful and without error message	
6	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor part at slide activated (-B4 = 1) then Conveyor motor -M1 variable speed (-Q3 = 1) switches on and stays on. The speed corresponds to the speed setpoint in the range +/- 50 rpm	
7	Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds)	
8	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0	
9	Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0	
10	Manual mode (-S0 = 0) → -Q3 = 0	
11	Switch off station (-K0 = 0) → -Q3 = 0	
12	Cylinder not retracted (-B1 = 0) → -Q3 = 0	
13	Speed > Motor_speed_monitoring_error_max → -Q3 = 0	
14	Speed < Motor_speed_monitoring_error_min → -Q3 = 0	
15	Project successfully archived	

9 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

[illegible]

Automation System SIMATIC S7-1500 SCE Training Curriculum

TIA Portal Module 0XX-600

Edition 04/2016



© Siemens AG 2016

TIA Portal Module 000-000 Concept- and Module Description	1
TIA Portal Module 012-100 Unspecified Hardware Configuration S7-1500	2
TIA Portal Module 012-101 Specified Hardware Configuration CPU 1516F-3 PN/DP	3
TIA Portal Module 012-105 Specified Hardware Configuration CPU 1512C-1 PN	4
TIA Portal Module 020-100 Process description of sorting station	5
TIA Portal Module 032-100 Basics of FC Programming	6
TIA Portal Module 032-200 Basics of FB Programming	7
TIA Portal Module 032-300 IEC Timers and IEC Counters	8
TIA Portal Module 032-410 Basics of Diagnostics	9
TIA Portal Module 032-420 Diagnostics via the Web	10
TIA Portal Module 032-500 Analog Values	11
TIA Portal Module 032-600 Global Data Blocks	12
TIA Portal Module 052-300 PID Controller	13

Matching SCE trainer packages for these training curriculums

- **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**
Order no.: 6ES7516-3FN00-4AB1
- **SIMATIC STEP 7 Professional V13 - Single license**
Order no.: 6ES7822-1AA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Classroom license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YA5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1AA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Upgrade license (up to 12 users)**
Order no.: 6ES7822-1BA03-4YE5
- **SIMATIC STEP 7 Professional V13 - Student license (up to 20 users)**
Order no.: 6ES7822-1AC03-4YA5

Please note that these trainer packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided at: [siemens.com/sce/tp](https://www.siemens.com/sce/tp)

Continued training

For regional Siemens SCE continued training, please contact your regional SCE contact
[siemens.com/sce/contact](https://www.siemens.com/sce/contact)

Additional information regarding SCE

[siemens.com/sce](https://www.siemens.com/sce)

Information regarding use

The SCE training curriculum for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems. This means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this training curriculum and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact: Roland Scheuerer
roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is expressly prohibited. We do not consent to commercial use of the training curriculums.

We wish to thank the TU Dresden, especially Prof. Dr.-Ing. Leon Urbas und Dipl.-Ing. Annett Pfeffer, the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

Table of contents

1	Goal	5
2	Prerequisite.....	5
3	Theory.....	5
3.1	Tasks of closed loop controls	5
3.2	Components of a control loop.....	6
3.3	Step function for analysis of controlled systems	8
3.4	Controlled systems with self-regulation.....	9
3.4.1	Proportional system without time delay	9
3.4.2	Proportional system with time delay	10
3.4.3	Proportional system with two time delays.....	11
3.4.4	Proportional system with n time delays	12
3.5	Systems without self-regulation.....	13
3.6	<i>Basic types of continuous controllers</i>	14
3.6.1	The proportional controller (P controller).....	15
3.6.2	The integral controller (I controller).....	17
3.6.3	The PI controller	18
3.6.4	The derivative controller (D controller)	19
3.6.5	The PID controller.....	19
3.7	Controller tuning using the oscillation test.....	20
3.8	Controller tuning with T_u - T_g approximation.....	21
3.8.1	Tuning the PI controller according to the Ziegler-Nichols method	22
3.8.2	Tuning the PI controller according to the Chien, Hrones and Reswick method	22
3.9	Digital controllers	23
4	Task	25
5	Planning.....	25
5.1	PID_Compact closed-loop control block.....	25
5.2	Technology diagram	26
5.3	Reference list.....	27

6	Structured step-by-step instructions	28
6.1	Retrieve an existing project	28
6.2	Call PID_Compact controller in a cyclic interrupt OB	30
6.3	Save and compile the program.....	37
6.4	Download the program	38
6.5	Monitor PID_Compact	39
6.6	PID_Compact pretuning	41
6.7	PID_Compact fine tuning.....	44
6.8	Archive the project	47
7	Checklist	48
8	Additional information	49

PID CONTROLLER FOR THE SIMATIC S7-1500

1 Goal

In this chapter, you will become acquainted with the use of software PID controllers for the SIMATIC S7-1500 with the TIA Portal programming tool.

The module explains the call-up, connection, configuration and optimization of a PID controller for the SIMATIC S7-1500. It also shows the steps for calling the PID controller in the TIA Portal and integrating it into a user program.

2 Prerequisite

This chapter builds on the chapter Analog Values with the SIMATIC S7 CPU1516F-3 PN/DP. You can use the following project for this chapter, for example: "SCE_EN_032-500_Analog_Values_R1508.zap13".

3 Theory

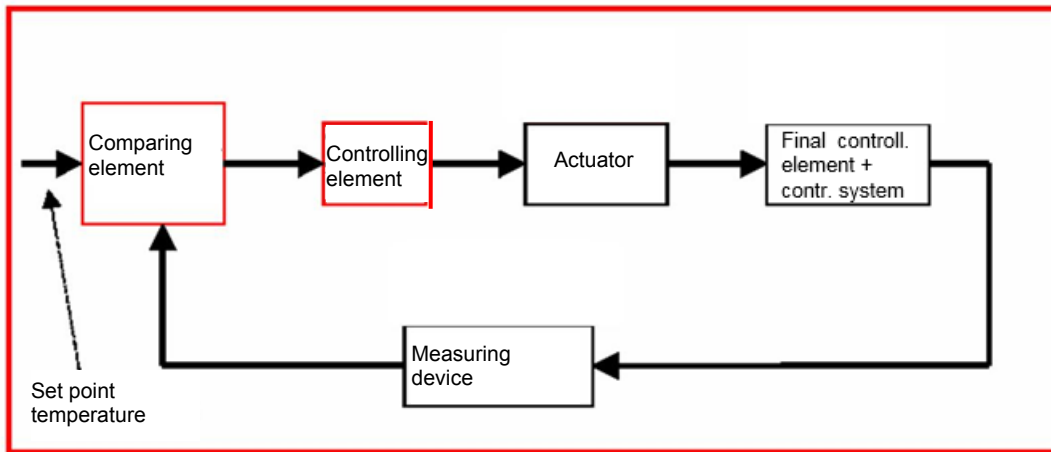
3.1 Tasks of closed loop controls

Closed loop control is a process in which the value of a variable is generated and maintained continuously through an intervention based on measurements of this variable.

This produces an action path that takes place in a closed loop – the control loop – because the process runs based on measurements of a variable that is, in turn, influenced by itself.

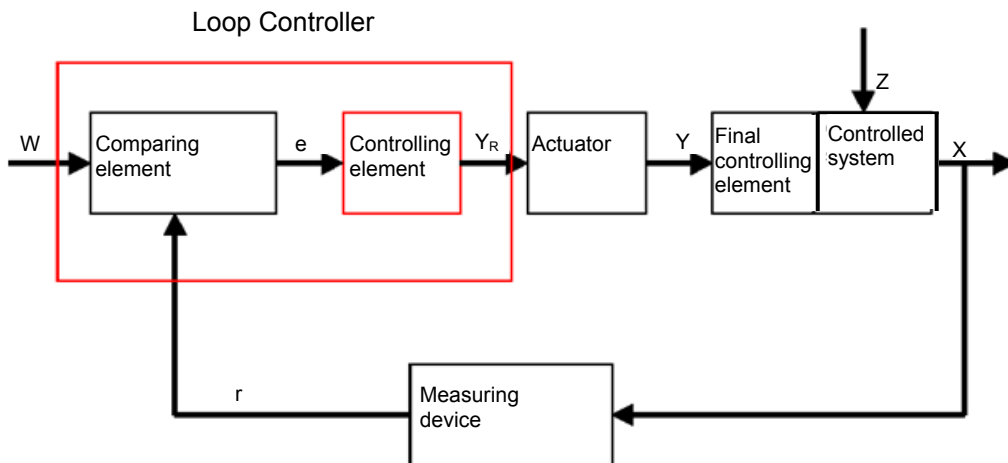
The variable to be controlled is continuously measured and compared with another preset variable of the same type. Depending on the result of this comparison, an adjustment of the variable to be controlled to the value of the preset variable is made.

Diagrammatic Representation of a Closed Loop Control



3.2 Components of a control loop

The fundamental concepts of closed loop controls are explained in detail in the following. An overview based on a diagram is presented here to start.



1. The controlled variable x

This is the actual "target" of the closed-loop control, namely the variable that is to be influenced or kept constant. In our example, this would be the room temperature. The instantaneous value of the controlled variable at a particular time is called the "actual value" at this time.

2. The feedback variable r

In a control loop, the controlled variable is continuously checked to enable a response to unwanted changes. The measured quantity proportional to the controlled variable is called the feedback variable. In the "Heating" example, it would correspond to the measured voltage of the inside thermometer.

3. The disturbance variable z

The disturbance variable is the variable that influences the controlled variable in an unwanted way and moves it away from the current setpoint. In the case of fixed setpoint control, this control is only necessary in the first place due to the existence of the disturbance variable. In the examined heating system, this would be, for example, the outside temperature or any other variable that causes the room temperature to move away from its ideal value.

4. The setpoint w

The setpoint at a given time is the value that the controlled variable should ideally have at this time. Note that the setpoint may vary continuously in a slave control. In our example, the setpoint would be the currently desired room temperature.

5. The comparing element

This is the point at which the current measured value of the controlled variable and the instantaneous value of the reference variable are compared. In most cases, both variables are measured voltages. The difference between the two variables is the "system error" e . This is passed to the controlling element and evaluated there (see below).

6. The controlling element

The controlling element is the actual heart of a closed loop control. It evaluates the system error, thus the information regarding whether, how and how much the controlled variable deviates from the current setpoint, as an input variable and derives from this the "**Controller output variable**" Y_R , which is ultimately used to influence the controlled variable. In the heating system example, the controller output variable would be the voltage for the mixer motor.

The manner in which the controlling element determines the controller output variable from the system error is the main criterion of the closed-loop control.

7. The actuator

The actuator is, so to speak, the "executive organ" of the closed loop control. It receives information from the controlling element in the form of the controller output variable indicating how the controlled variable is to be influenced and translates this into a change of the "manipulated variable". In our example, this would be the mixer motor controller.

8. The final controlling element

This is the element of the control loop that influences the controlled variable (more or less directly) as a function of the **manipulated variable** Y . In the example, this would be the combination of the mixer, heating lines and radiators. The adjustment of the mixer (the manipulated variable) is made by the mixer motor (actuator) and influences the room temperature by means of the water temperature.

9. The controlled system

The controlled system is the system containing the variable to be controlled, thus the living space in the heating example.

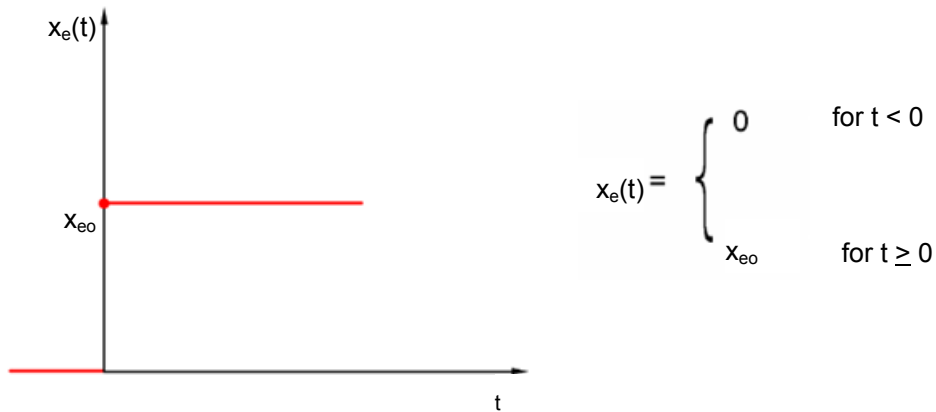
10. The dead time

The dead time refers to the time that elapses from a change in the controller output variable until there is a measurable response in the controlled system. In the example, this would be the time between a change in the voltage for the mixer motor and a measurable change in the room temperature resulting from this.

3.3 Step function for analysis of controlled systems

To analyze the response of controlled systems, controllers and control loops, a uniform function for the input signal is used – the step function.

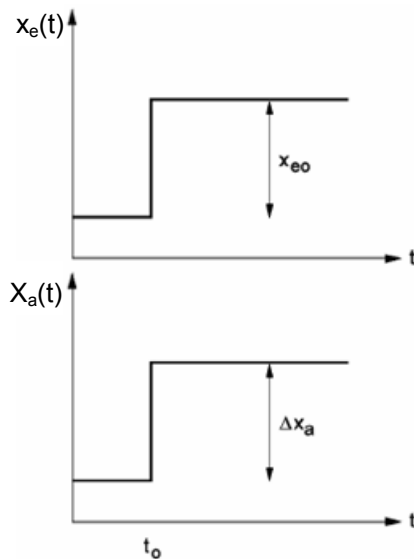
Depending on whether a control loop element or the entire control loop is being analyzed, the controlled variable $x(t)$, the manipulated variable $y(t)$, the reference variable $w(t)$ or the disturbance variable $z(t)$ can be assigned the step function. The input signal is often designated $x_e(t)$ and the output signal $x_a(t)$.



3.4 Controlled systems with self-regulation

3.4.1 Proportional system without time delay

This controlled system is called a P system for short.



sudden change of the input variable at t_0

Controlled variable/manipulated variable:

$$x = K_{ss} \cdot y$$

K_{ss} : Proportional coefficient for a manipulated variable change:

$$K_{ss} = \frac{\Delta x}{\Delta y} = \tan \alpha$$

Controlled variable/disturbance variable:

$$x = K_{sz} \cdot z$$

K_{sz} : Proportional value for a disturbance variable change

Range:

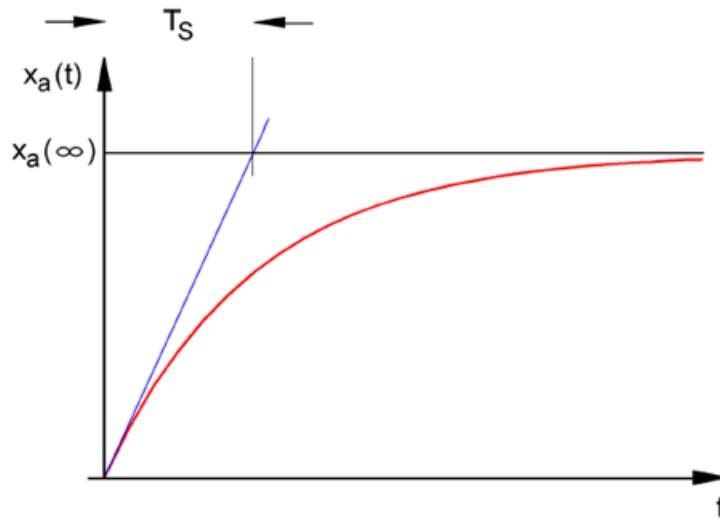
$$y_h = y_{\max} - y_{\min}$$

Control range:

$$x_h = x_{\max} - x_{\min}$$

3.4.2 Proportional system with time delay

This controlled system is called a P-T1 system for short.



Differential equation for a general input signal $x_e(t)$:

$$T_S \cdot \dot{x}_a(t) + x_a(t) = K_{PS} \cdot x_e(t)$$

Solution of the differential equation for a step function at the input (step response)

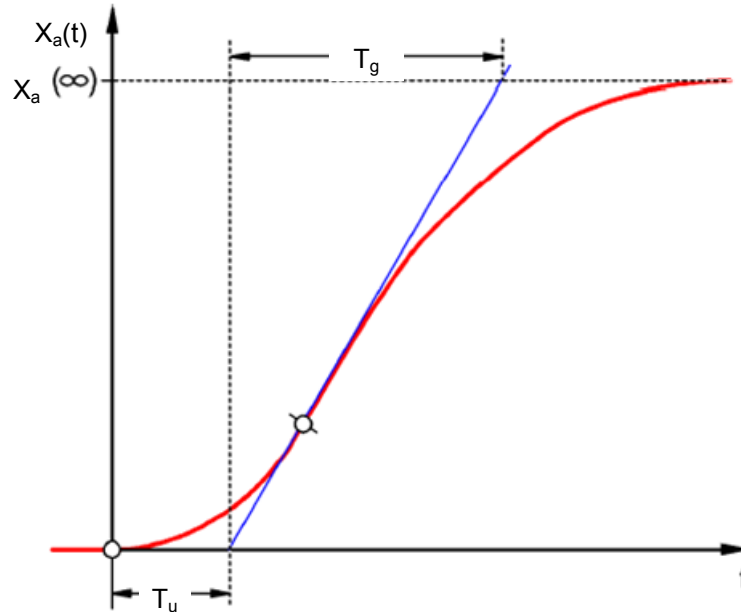
$$x_a(t) = K_{PS} (1 - e^{-t/T_S}) \cdot x_{e0}$$

$$x_a(t = \infty) = K_{PS} \cdot x_{e0}$$

T_S : Time constant

3.4.3 Proportional system with two time delays

This system is called a P-T2 system for short.



Tu: Delay time Tg: Compensation time

The system is generated through the reaction-free series connection of two P-T1 systems that have the time constants TS1 and TS2.

Controllability of P-Tn systems:

$$\frac{T_u}{T_g} < \frac{1}{10} \rightarrow \boxed{\text{easily controllable}} \quad \frac{T_u}{T_g} \approx \frac{1}{6} \rightarrow \boxed{\text{still controllable}} \quad \frac{T_u}{T_g} > \frac{1}{3} \rightarrow \boxed{\text{difficult to control}}$$

With the increasing ratio T_u/T_g , the system becomes less and less controllable.

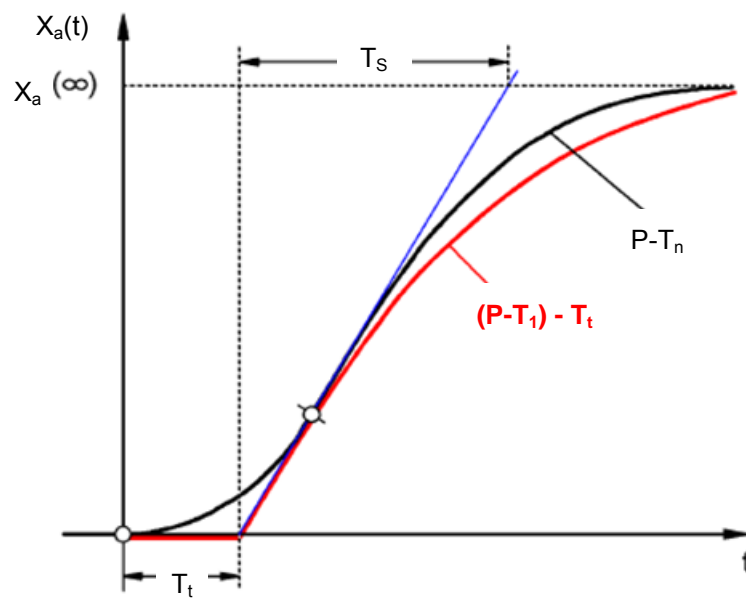
3.4.4 Proportional system with n time delays

This controlled system is called a P-T_n system for short.

The time response is described by an n th order differential equation. The step response characteristic is similar to that of the P-T₂ system. The time response is described by T_u and T_g .

Substitute: An approximate substitution for the system with many delays is the series connection of a P-T₁ system with a dead time system.

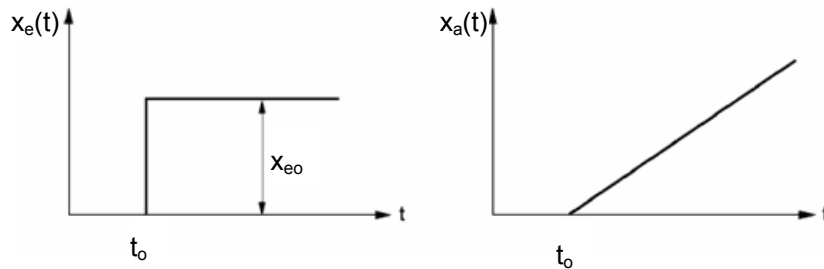
The following applies: $T_t \gg T_u$ and $T_S \gg T_g$.



3.5 Systems without self-regulation

This controlled system is called an I system for short.

After a disturbance, the controlled variable continues increasing steadily without striving for a fixed final value.

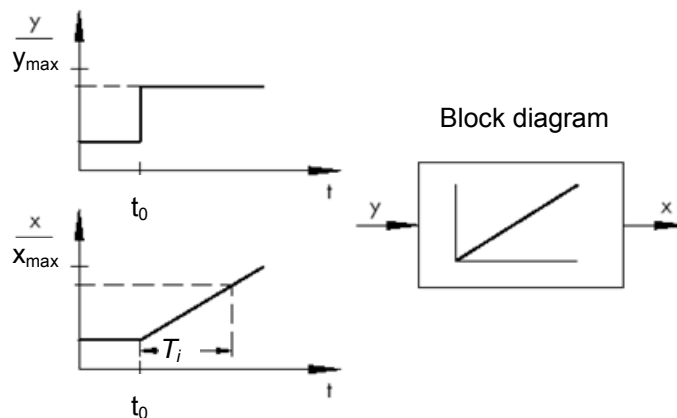


Example: Level control

For a tank with discharge outlet, whose incoming and outgoing flow rates are the same, there is a constant fill height. If the incoming or outgoing flow rate changes, the liquid level rises or falls. The level changes faster as the difference between the incoming flow rate and outgoing flow rate increases.

It is clear from this example that, in practice, the integral action has a limit in most cases. The controlled variable increases or decreases only until a system-inherent limit value is reached. A tank runs over or drains dry, pressure reaches the system maximum or minimum, etc.

The figure shows the time response of an I system to a step change in the input variable as well as the derived block diagram:



If the step function at the input changes to a function $x_e(t)$, then

$$x_a(t) = K_{IS} \int x_e(t) dt \Rightarrow \text{integrating controlled system}$$

K_{IS} : Integral coefficient of the controlled system

* Figure from SAMSON Technical Information - L102 Controllers and Controlled Systems, Edition: August 2000 (http://www.samson.de/pdf_en/l102en.pdf)

3.6 *Basic types of continuous controllers*

Discrete controllers that only switch one or two manipulated variables on and off have the advantage of simplicity. Both the controller itself and the actuator and final controlling element are simpler in nature and thus less expensive than continuous controllers.

Discrete controllers have several disadvantages, however. For one thing, when large loads such as large electric motors or cooling units must be switched, high load peaks may occur at switch-on and overload the power supply, for example. For this reason, these often do not switch between "Off" and "On" but instead between full power ("full load") and a significantly lower power of the actuator or final controlling element ("base load"). Still, even with this improvement, a discrete closed-loop control is unsuitable for numerous applications. Consider an automobile engine whose speed is discretely controlled. There would then be nothing between idle and full throttle. Apart from the fact that it would probably be impossible to properly transfer the forces from a sudden full-throttle to the road via the tires, such a vehicle would probably be unsuitable for road traffic.

Continuous controllers are therefore used for such applications. Theoretically, hardly any limits are placed on the mathematical relationship that establishes the controlling element between the system error and controller output variable. In practice, however, three classic basic types are differentiated. These will be described in more detail in the following.

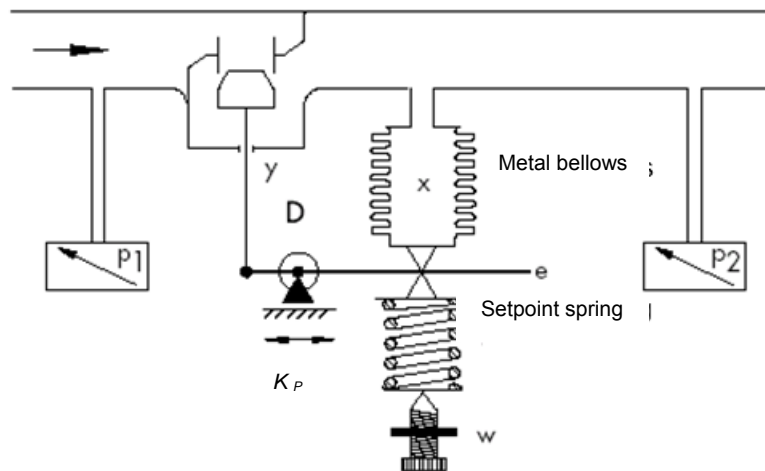
3.6.1 The proportional controller (P controller)

The manipulated variable y of a P controller is proportional to the measured error e . From this can be deduced that a P controller reacts to any deviation without lag and only generates a manipulated variable in case of system deviation.

The proportional pressure controller illustrated in the figure compares the force F_S of the setpoint spring with the force F_B created in the elastic metal bellows by the pressure p_2 . When the forces are off balance, the lever pivots about point D. This changes the position of the valve plug –and, hence, the pressure p_2 to be controlled –until a new equilibrium of forces is restored.

The dynamic behavior of the P controller after a step change in the error variable is shown in the figure. The amplitude of the manipulated variable y is determined by the error e and the proportional-action coefficient K_P .

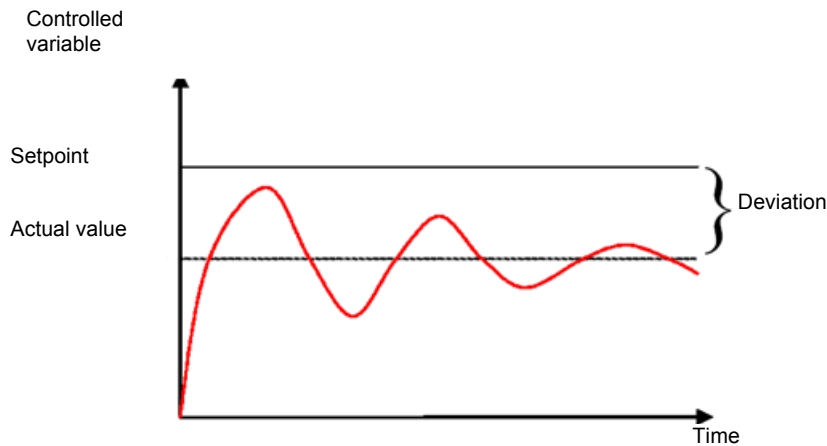
To keep the control deviation as small as possible, as large a proportional-action coefficient as possible must be selected. An increase in the factor causes the controller to react faster, but if the value is too high there is a risk of overshooting and a large "hunting" tendency of the controller.



$$y = K_P \cdot e$$

* Figure and text from SAMSON Technical Information - L102 Controllers and Controlled Systems, Edition: August 2000 (http://www.samson.de/pdf_en/l102en.pdf)

You see the response of the P controller in the diagram.



The advantages of this controller type lie, on the one hand, in its simplicity (in the simplest case, it can be implemented electronically with just a resistor) and, on the other hand, in its very prompt reaction compared to other controller types.

The main disadvantage of the P controller is its permanent system deviation. That is, the setpoint is never fully reached even over the long term. This disadvantage as well as the not yet ideal response speed cannot be minimized to a satisfactory extent through a larger proportional-action coefficient, because this leads to overshooting by the controller, or in other words an overreaction. In the worst case, the controller goes into a permanent oscillation in which the controlled variable is periodically moved away from the setpoint by the controller itself instead of by the manipulated variable.

The problem of permanent control deviation is best solved by an additional integral controller.

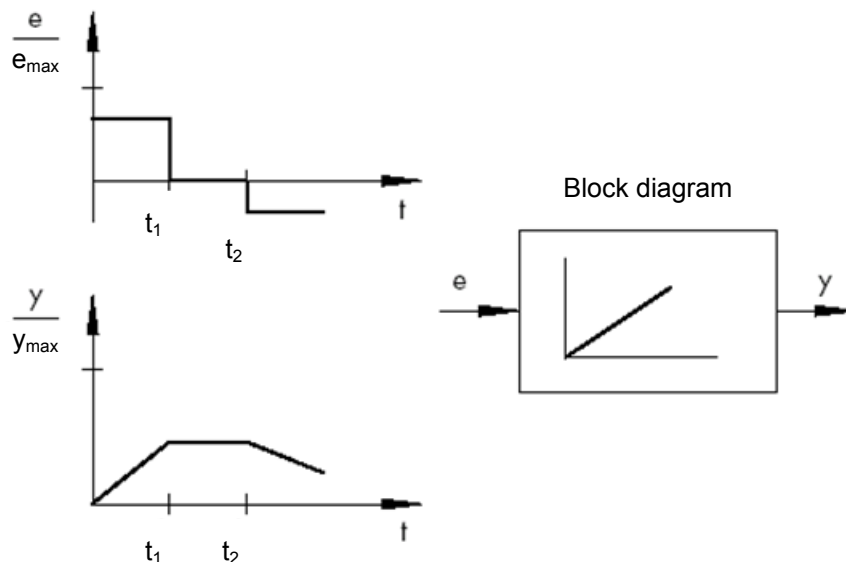
3.6.2 The integral controller (I controller)

Integral control action is used to fully correct system deviations at any operating point. As long as the error is nonzero, the integral action will cause the value of the manipulated variable to change. Only when reference variable and controlled variable are equally large –at the latest, though, when the manipulated variable reaches its system specific limit value (Umax, pmax, etc.)– is the control process balanced.

Mathematics expresses integral action as follows: the value of the manipulated variable is changed proportional to the integral of the error e.

$$y = K_i \int e \, dt \quad \text{with} \quad K_i = \frac{1}{T_n}$$

How rapidly the manipulated variable increases/decreases depends on the error and the integral time.

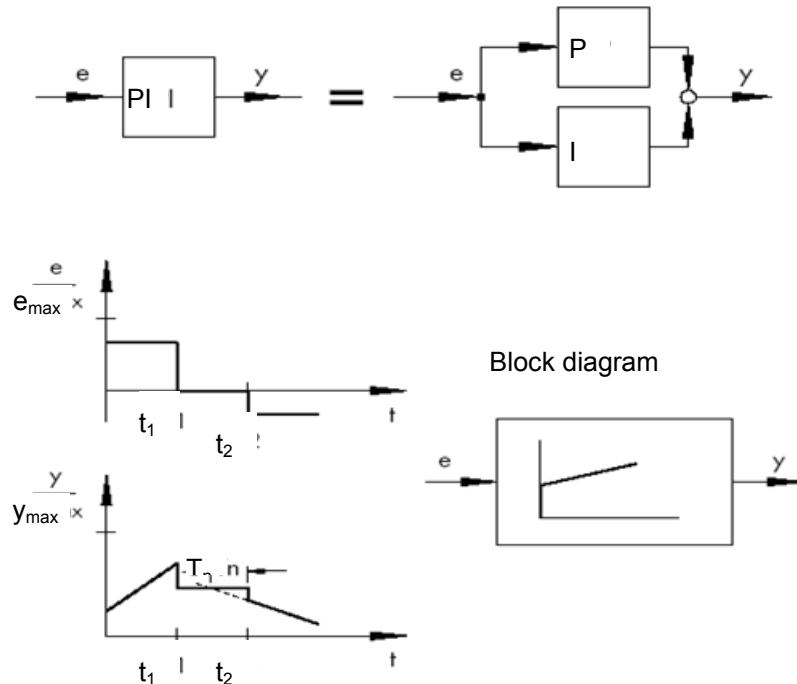


* Figure and text from SAMSON Technical Information - L102 Controllers and Controlled Systems, Edition: August 2000 (http://www.samson.de/pdf_en/l102en.pdf)

3.6.3 The PI controller

PI controllers are often employed in practice. In this combination, one P and one I controller are connected in parallel.

If properly designed, they combine the advantages of both controller types (stability and rapidity; no steady-state error), so that their disadvantages are compensated for at the same time.



The dynamic behavior is marked by the proportional-action coefficient K_p and the reset time T_n . Due to the proportional component, the manipulated variable immediately reacts to any error signal e , while the integral component starts gaining influence only after some time. T_n represents the time that elapses until the I component generates the same control amplitude that is generated by the P component (K_p) from the start. As with I controllers, the reset time T_n must be reduced if the integral-action component is to be amplified.

Controller dimensioning:

By adjusting the K_p and T_n values, oscillation of the controlled variable can be reduced, however, at the expense of control dynamics.

PI controller applications: Fast control loops allowing no steady-state error

Examples: pressure, temperature, ratio control, etc.

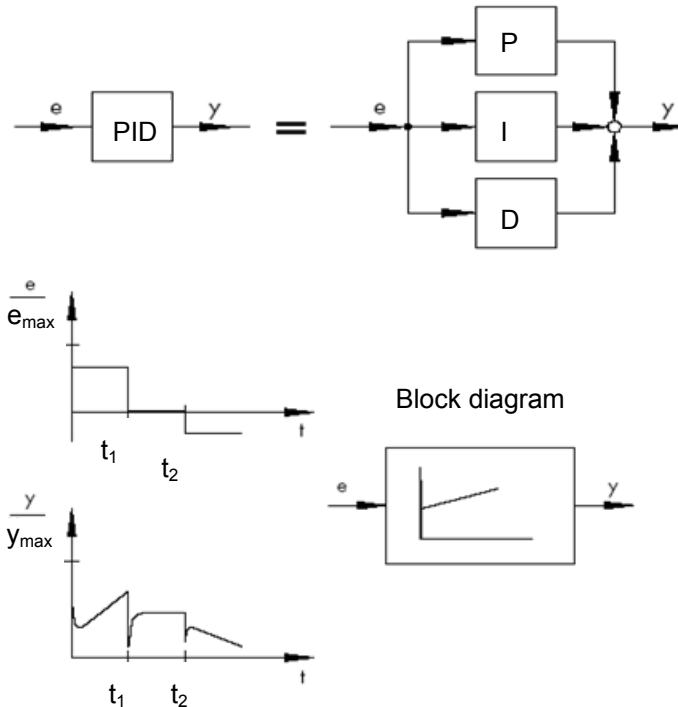
* Figure and text from SAMSON Technical Information - L102 Controllers and Controlled Systems, Edition: August 2000 (http://www.samson.de/pdf_en/l102en.pdf)

3.6.4 The derivative controller (D controller)

D controllers generate the manipulated variable from the rate of change of the error and not – as P controllers – from their amplitude. Therefore, they react much faster than P controllers: even if the error is small, derivative controllers generate – by anticipation, so to speak – large control amplitudes as soon as a change in amplitude occurs. A steady-state error signal, however, is not recognized by D controllers, because regardless of how big the error, its rate of change is zero. Therefore, derivative-only controllers are rarely used in practice. They are usually found in combination with other control elements, mostly in combination with proportional control.

3.6.5 The PID controller

If a D component is added to PI controllers, the result is an extremely versatile PID controller. As with PD controllers, the added D component – if properly tuned – causes the controlled variable to reach its setpoint more quickly, thus reaching steady state more rapidly.



$$y = K_p \cdot e + K_i \int e \, dt + K_D \frac{de}{dt} \quad \text{with} \quad K_i = \frac{K_p}{T_n}; \quad K_D = K_p \cdot T_v$$

* Figure and text from SAMSON Technical Information - L102 Controllers and Controlled Systems, Edition: August 2000 (http://www.samson.de/pdf_en/l102en.pdf)

3.7 Controller tuning using the oscillation test

For a satisfactory control result, the selection of a suitable controller is an important aspect. It is even more important that the control parameters K_p , T_n and T_v be appropriately adjusted to the system response. Mostly, the adjustment of the controller parameters remains a compromise between a very stable, but also very slow control loop and a very dynamic, but irregular control response which may easily result in oscillation, making the control loop unstable in the end.

For nonlinear systems that should always work in the same operating point, e.g. fixed setpoint control, the controller parameters must be adapted to the system response at this particular operating point. If a fixed operating point cannot be defined, such as with follow-up control systems \ddot{n} , the controller must be adjusted to ensure a sufficiently rapid and stable control result within the entire operating range.

In practice, controllers are usually tuned on the basis of values gained by experience.

Should these not be available, however, the system response must be analyzed in detail, followed by the application of several theoretical or practical tuning approaches in order to determine the proper control parameters.

One approach is a method first proposed by Ziegler and Nichols, the so-called ultimate method. It provides simple tuning that can be applied in many cases. This method, however, can only be applied to controlled systems that allow sustained oscillation of the controlled variable.

For this method, proceed as follows:

- At the controller, set K_p and T_v to the lowest value and T_n to the highest value (smallest possible influence of the controller).
- Adjust the controlled system manually to the desired operating point (start up control loop).
- Set the manipulated variable of the controller to the manually adjusted value and switch to automatic operating mode.
- Continue to increase K_p (decrease X_p) until the controlled variable encounters harmonic oscillation. If possible, small step changes in the setpoint should be made during the K_p adjustment to cause the control loop to oscillate.
- Take down the adjusted K_p value as critical proportional-action coefficient $K_{p,crit}$. Determine the time span for one full oscillation amplitude as T_{crit} , if necessary by taking the time of several oscillations and calculating their average.
- Multiply the values of $K_{p,crit}$ and T_{crit} by the values according to the table and enter the determined values for K_p , T_n and T_v at the controller.

	K_p	T_n	T_v
P	$0.50 \times K_{p, crit.}$	-	-
PI	$0.45 \times K_{p, crit.}$	$0.85 \times T_{crit.}$	-
PID	$0.59 \times K_{p, crit.}$	$0.50 \times T_{crit.}$	$0.12 \times T_{crit.}$

* Figure and text from SAMSON Technical Information - L102 Controllers and Controlled Systems, Edition: August 2000 (http://www.samson.de/pdf_en/l102en.pdf)

3.8 Controller tuning with T_u - T_g approximation

The tuning of the controlled systems will be performed here using the example of a PT2 system.

T_u - T_g approximation

The Ziegler-Nichols method and the Chien, Hrones and Reswick method are based on the T_u - T_g approximation in which the transfer coefficient of the system K_s , delay time T_u and balancing time T_g parameters are determined from the system step response.

The tuning rules, which are described below, are the result of experiments using analog computer simulations.

P- T_N systems can be described with sufficient accuracy with a so-called T_u - T_g approximation, that is, through approximation using a P- T_1 - T_L system.

The starting point is the system step response with input step height K . The required parameters (transfer coefficient of the system K_s , delay time T_u and balancing time T_g) are determined as shown in the figure.

The transfer function must be measured up to the final steady-state value ($K \cdot K_s$) so that the transfer coefficient of the system K_s required for the calculation can be determined.

The main advantage of this method is that the approximation can also be used when an analytical description of the system is not possible.

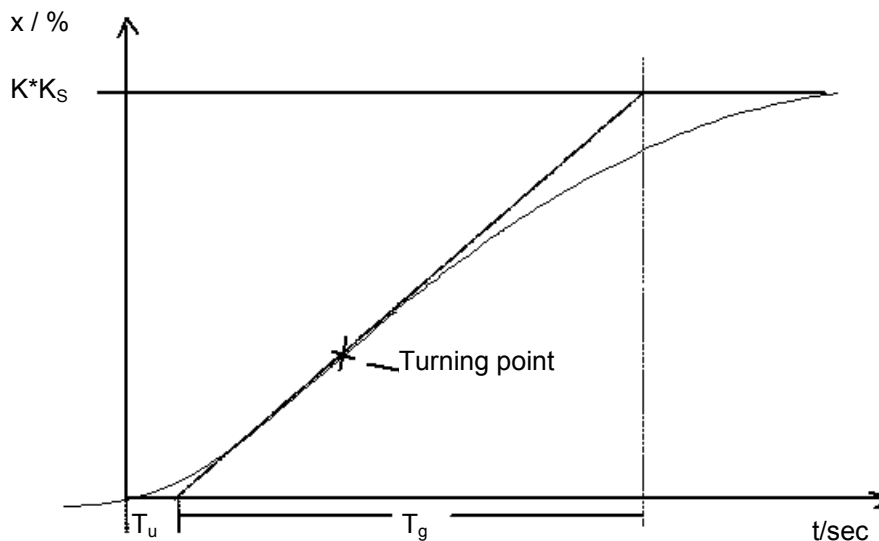


Figure: T_u - T_g -Approximation

3.8.1 Tuning the PI controller according to the Ziegler-Nichols method

Based on experiments on P-T₁-T_L systems, Ziegler and Nichols have identified the following optimal controller adjustments for fixed setpoint control:

$$K_{PR} = 0,9 \frac{T_g}{K_S T_u}$$

$$T_N = 3,33 T_u$$

Use of these tuning values generally results in very good response to disturbances.

3.8.2 Tuning the PI controller according to the Chien, Hrones and Reswick method

Both the response to disturbances and response to setpoint changes were examined in order to achieve the most favorable controller parameters. Different values are yielded for the two cases. In addition, two different adjustments are specified in each case that meets different control performance requirements.

This resulted in the following adjustments:

- For response to disturbances:

Aperiodic transient reaction
with the shortest duration

20% overshoot minimum
oscillation period

$$K_{PR} = 0,6 \frac{T_g}{K_S T_u}$$

$$K_{PR} = 0,7 \frac{T_g}{K_S T_u}$$

$$T_N = 4 T_u$$

$$T_N = 2,3 T_u$$

- For response to setpoint changes:

Aperiodic transient reaction
with the shortest duration

20 % overshoot minimum
oscillation period

$$K_{PR} = 0,35 \frac{T_g}{K_S T_u}$$

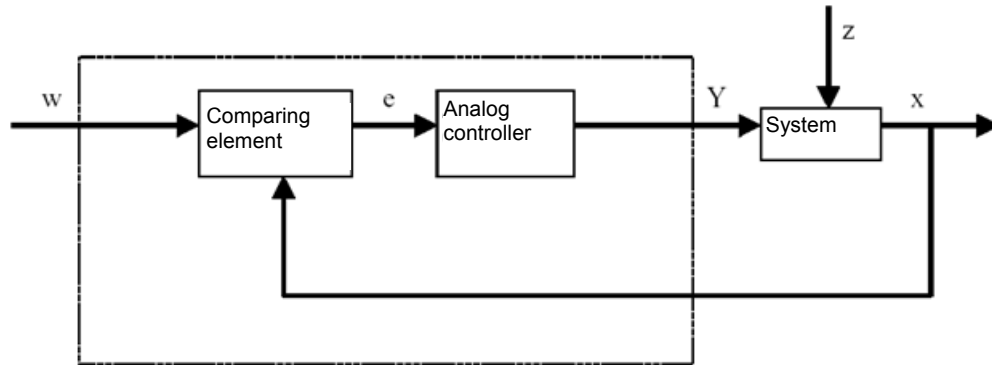
$$K_{PR} = 0,6 \frac{T_g}{K_S T_u}$$

$$T_N = 1,2 T_g$$

$$T_N = T_g$$

3.9 Digital controllers

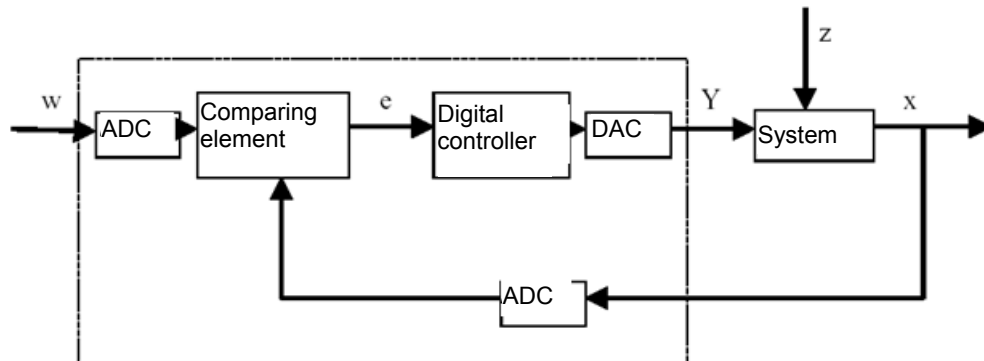
Up to now, the main focus was on analog controllers, in other words, controllers that use the system error, which exists as an analog value, to derive the controller output variable in an analog manner. The diagram of this type of control loop is now well-known:



Often, however, it is advantageous to perform the actual evaluation of the system error digitally. For one thing, the relationship between the system error and controller output variable can be defined much more flexibly when it can be defined by an algorithm or formula that can be used in each case to program a computer than when it has to be implemented in the form of an analog circuit. For another, digital technology enables significantly greater integration of circuits so that multiple controllers can be accommodated in the smallest space. Finally, by dividing the computing time when there is a sufficient amount of computing capacity, it is even possible to use an individual computer as a controller for multiple control loops.

To enable digital processing of the variables, both the reference variable and the feedback variable are first converted to digital values in an analog-to-digital converter (ADC). These are then subtracted from one another by a digital comparing element and the difference is passed to the digital controlling element. Its controller output variable is then converted back to an analog value in a digital-to-analog converter (DAC). From the outside, the combined unit of converters, comparing element and controlling element resembles an analog controller.

We will examine the structure of a digital controller based on a diagram:



The advantages resulting from digital implementation of the controller are accompanied by various problems. For this reason, the size of some variables related to the digital controller must be chosen large enough to prevent the accuracy of the closed loop control from suffering too much from digitization.

Quality criteria for digital computers are:

- The quantization resolution of the digital-to-analog converter

This specifies how fine the continuous value range is digitally mapped. The chosen resolution must be high enough that none of the finer points important for the closed loop control are lost.

- The sampling rate of the analog-to-digital converter.

This is the frequency at which the analog values present at the converter are measured and digitized. This must be high enough that the controller can also still respond to step changes in the controlled variable in a timely manner.

- The cycle time

Unlike an analog closed-loop controller, each digital computer works in clock cycles. The speed of the utilized computer must be high enough that a significant change of the controlled variable cannot occur during a single clock cycle (in which the output value is calculated and no input value is queried).

The performance of the digital controller must be high enough that its response is apparently as prompt and precise as an analog controller.

4 Task

In this chapter, a PID controller for speed control will be added to the program from chapter "SCE_EN_032-500 Analog Values". The call-up of the "MOTOR_SPEEDCONTROL" [FC10] function must be deleted for this.

5 Planning

The PID_Compact technology object is available in the TIA Portal for closed loop controls.

For closed-loop control of the motor speed, this technology object replaces the "MOTOR_SPEEDCONTROL" [FC10] block.

This will be carried out as an expansion of the "032-500_Analog_Values_" project. This project must be retrieved from the archive beforehand.

The call-up of the "MOTOR_SPEEDCONTROL" [FC10] function must be deleted in the "Main" [OB1] organization block before the technology object can be called and connected in a cyclic interrupt OB.

The PID_Compact technology object must then be configured and commissioned.

5.1 PID_Compact closed-loop control block

The PID_Compact technology object provides a PID controller with integrated tuning for proportional-action final controlling elements.

The following operating modes are possible:

- Inactive
- Pretuning
- Fine tuning
- Automatic mode
- Manual mode
- Substitute output value with error monitoring

Here, the connection, parameter assignment and commissioning of this controller will be for automatic mode

During commissioning we will use the integrated tuning algorithms and record the control response of the controlled system.

The PID_Compact technology object is always called from a cyclic interrupt OB whose fixed set cycle time is 50 ms here.

The speed setpoint is set as a constant at the "Setpoint" input of the PID_Compact technology object in revolutions per minute (range: +/- 50 rpm). The data type is 32-bit floating-point number (Real).

The actual speed value -B8 (sensor actual value speed of the motor +/-10V corresponds to +/- 50 rpm) will be entered at the "Input_PER" input.

The output of the controller "Output_PER" will then be connected directly with signal -U1 (manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm).

The controller will only be active as long as output -Q3 (conveyor motor -M1 variable speed) is set. If this is not set, the controller will be deactivated by connection of the "Reset" input.

5.2 Technology diagram

Here you see the technology diagram for the task.

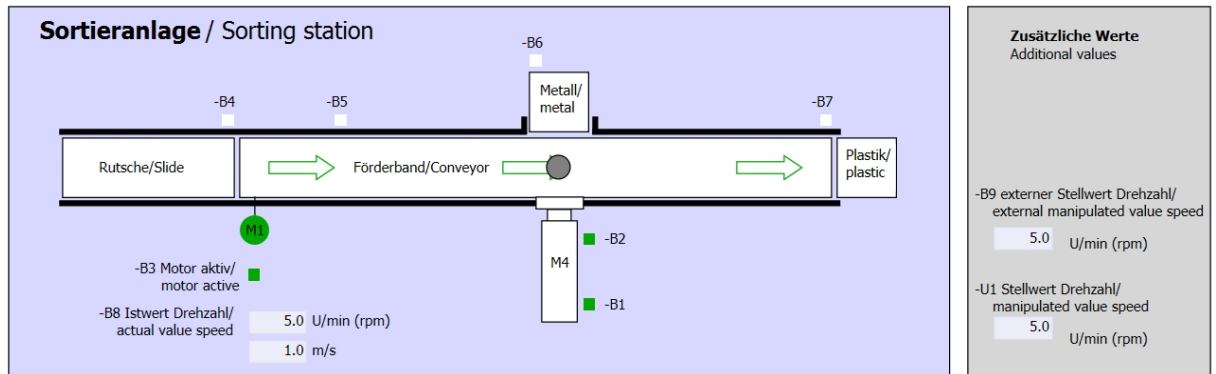


Figure 1: Technology diagram

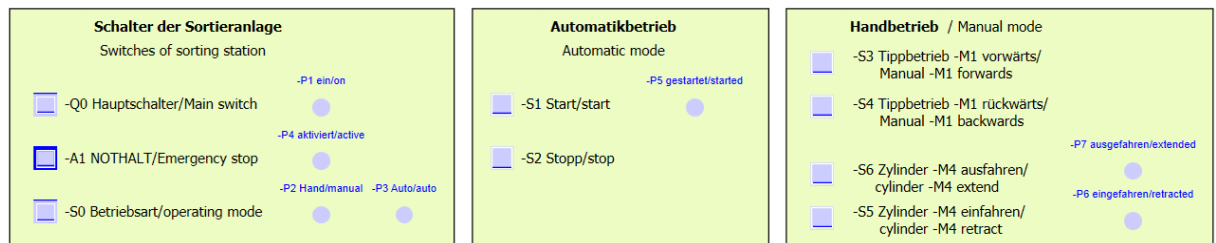


Figure 2: Control panel

5.3 Reference list

The following signals are required as global operands for this task.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop OK	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	-S1	Pushbutton automatic start	NO
I 0.4	BOOL	-S2	Pushbutton automatic stop	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO
I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO
IW64	BOOL	-B8	Sensor actual value speed of the motor +/-10V corresponds to +/- 50 rpm	

DO	Type	Identifier	Function	
Q 0.2	BOOL	-Q3	Conveyor motor -M1 variable speed	
QW 64	BOOL	-U1	Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm	

Legend for reference list

DI Digital Input

AI Analog Input

I Input

NC Normally Closed

NO Normally Open

DO Digital Output

AO Analog Output

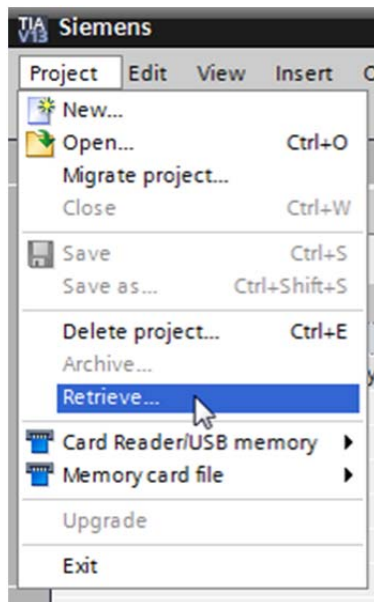
Q Output

6 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

6.1 Retrieve an existing project

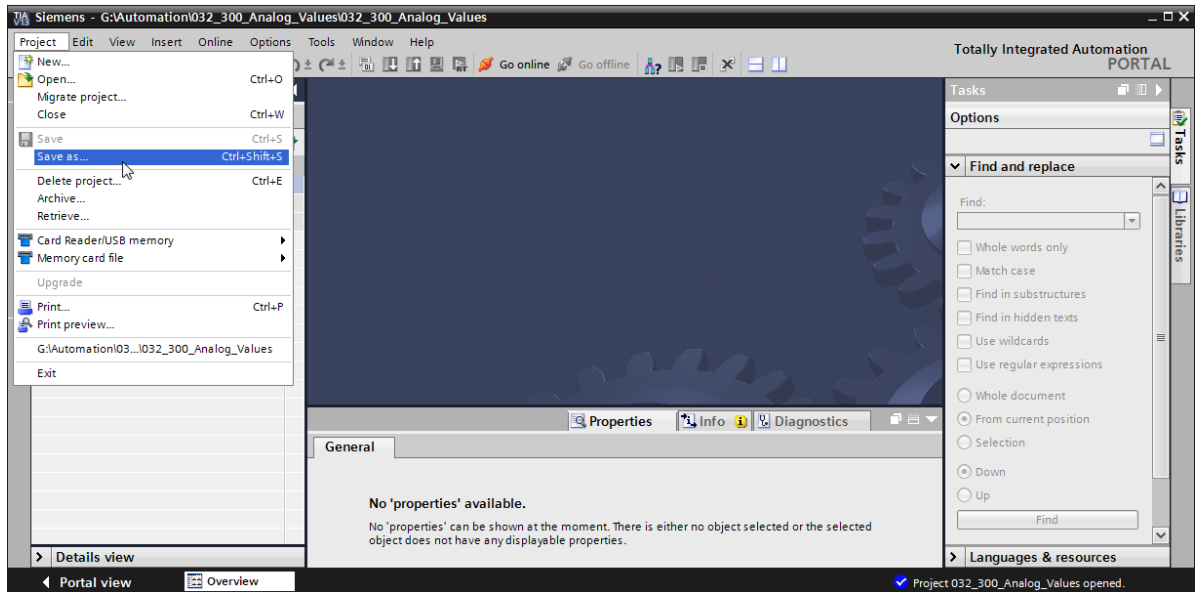
- Before we can expand the "SCE_EN_032-500_Analog_Values_R1508.zap13" project from chapter "SCE_EN_032-500_Analog_Values", we must retrieve this project from the archive. To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open.
(→ Project → Retrieve → Select a .zap archive → Open)



- The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK".
(→ Target directory → OK)

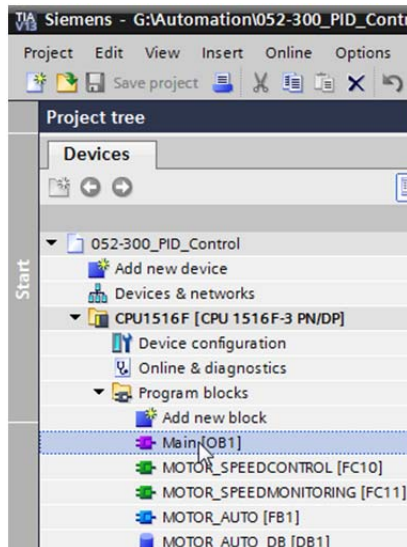
→ Save the opened project under the name 052-300_PID_Controller.

(→ Project → Save as ... → 052-300_PID_Controller → Save)



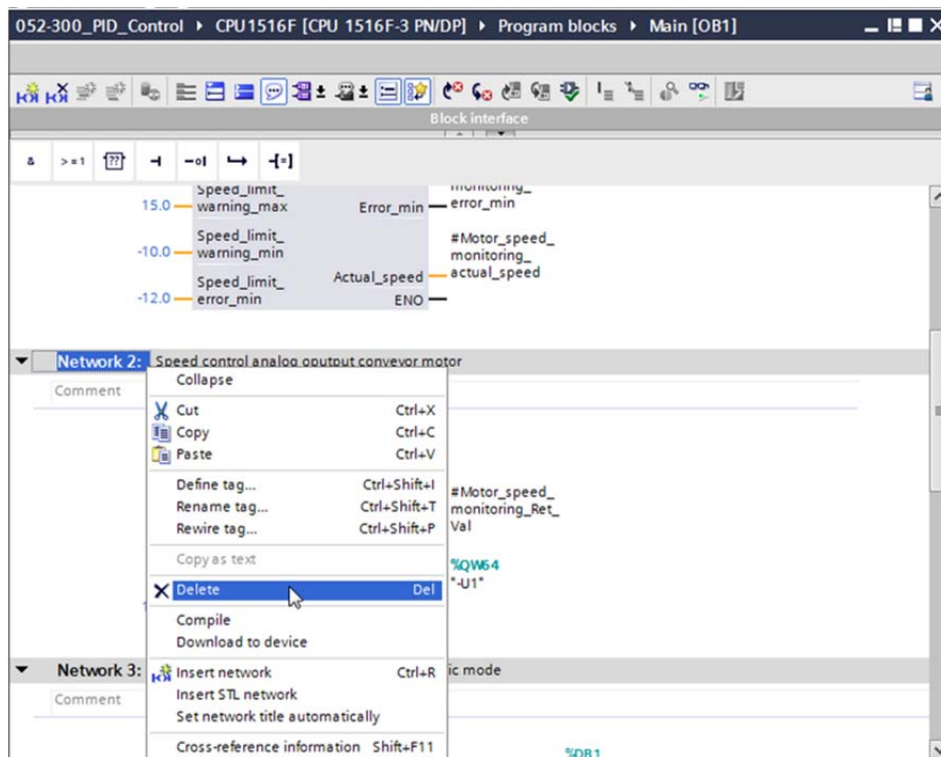
6.2 Call PID_Compact controller in a cyclic interrupt OB

→ Open the "Main [OB1]" organization block with a double-click.



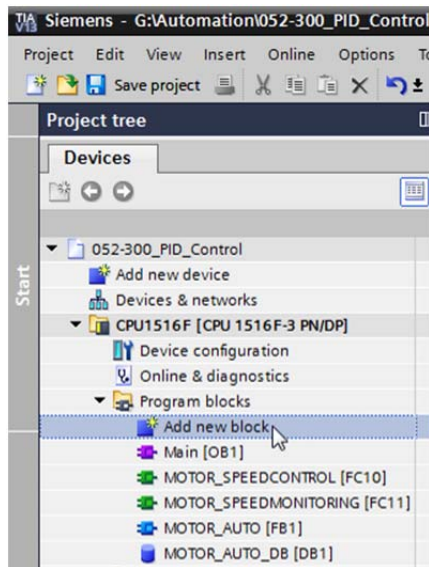
→ Delete Network 2 with the no longer needed call-up of the "MOTOR_SPEEDCONTROL" [FC10] function.


(→ Network 2 → Delete)




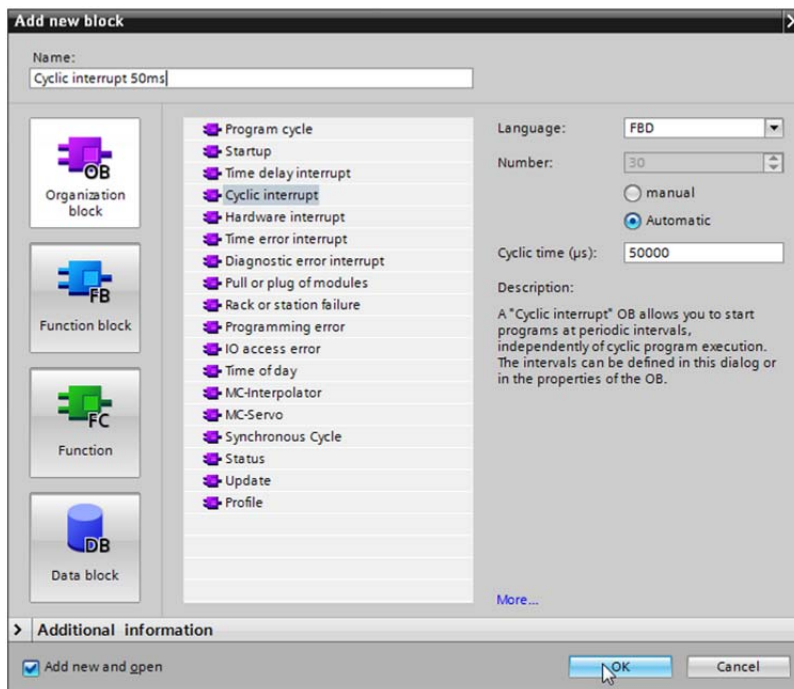
- We need a cyclic interrupt OB for calling the PID_Compact controller. Therefore, select the 'Add new block' item in the Program blocks folder.

(→ Program blocks → Add new block)

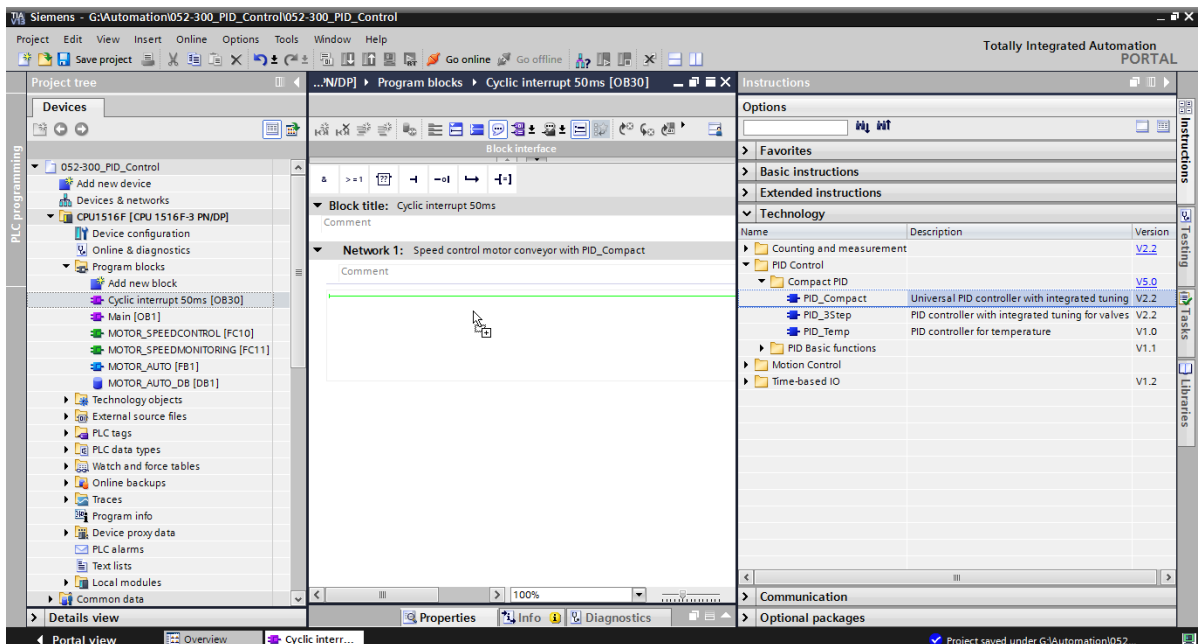


- Select  in the next dialog and rename the cyclic interrupt OB to: "Cyclic interrupt 50ms". Set the language to FBD and assign "50000 µs" as the cyclic time. Select the "Add new and open" check box. Click "OK".

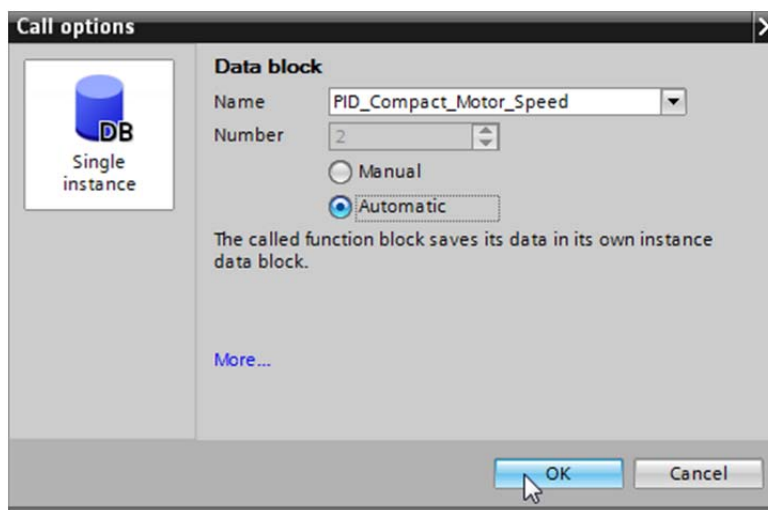
(→  → Name: Cyclic interrupt 50ms → Language: FBD → Cyclic time (ms): 50000 → ☒ Add new and open → OK)







- The block is then directly opened. Enter meaningful comments and move the 'PID_Compact' technology object to Network 1 using drag-and-drop.
- (→ Technology → PID Control → Compact PID → PID_Compact)

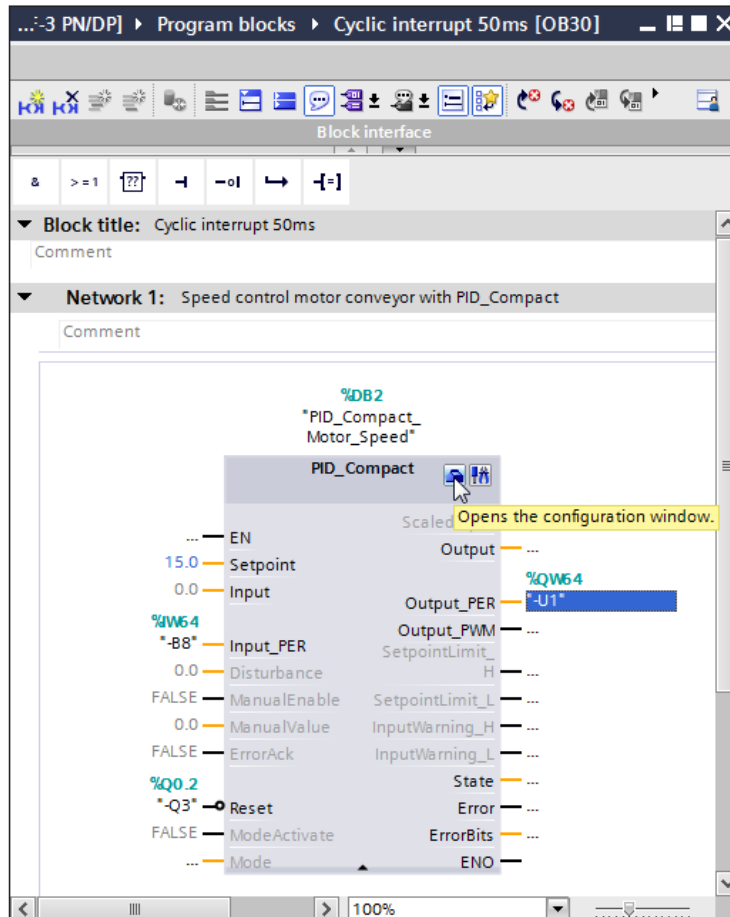


- Assign a name for the instance data block and apply it with OK.
- (→ PID_Compact_Motor_Speed → OK)



- Expand the view of the block by clicking the '▲' arrow. Interconnect this block as shown here with setpoint (constant: 15.0), actual value (global tag "-B8"), manipulated variable (global tag "-U1") and Reset input for deactivating the controller (global tag "-Q3"). Negate the 'Reset' input. The configuration mask  of the controller can then be opened.

(→  → 15.0 → "-B8" → "-U1" → -Q3 →  → )



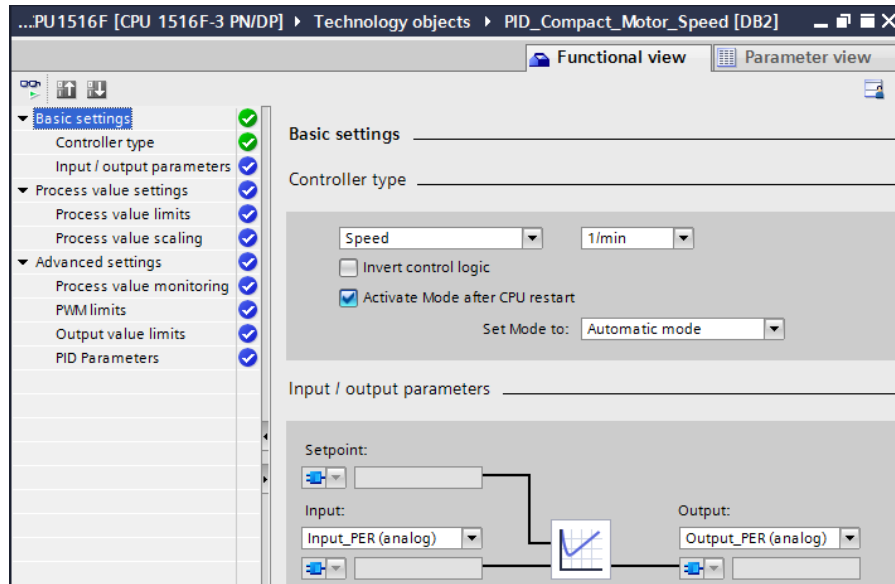
- There are 2 views for configuration of the controller: Parameter view and Functional view. Here we will use the easier-to-understand 'Functional view'.

(→ Functional view)

052-300_PID_Control ▸ CPU1516F [CPU 1516F-3 PN/DP] ▸ Technology objects ▸ PID_Compact_Motor_Speed [DB2]						
Functional view						
Functional navigators < no text filter >						
	Name in functional view	Name in DB	Start value project	Minimum value	Maximum value	Comment
▼ All parameters						
▼ Configuration parameters						
▼ Basic settings						
Controller type	Physical quantity	PhysicalQuantity	✓ Speed			Selection of physical quantity.
Input / output parameters	Physical quantity	PhysicalQuantity	✓ 17			Selection of physical quantity.
Process value settings	Unit of measurement	PhysicalUnit	✓ 1/min			Selection of unit of measureme...
Advanced settings	Unit of measurement	PhysicalUnit	✓ 0			Selection of unit of measureme...
Commissioning parameters	Invert control logic	..InvertControl	✓ FALSE			Enables inversion of control logic
Other parameters	Activate Mode after CPU restart	RunModeByStartup	✓ TRUE			Activates the operating mode s...
	Set Mode to	Mode	✓ Automatic mode	0	4	Selection of operating mode.
		Mode	✓ 3			Selection of operating mode.

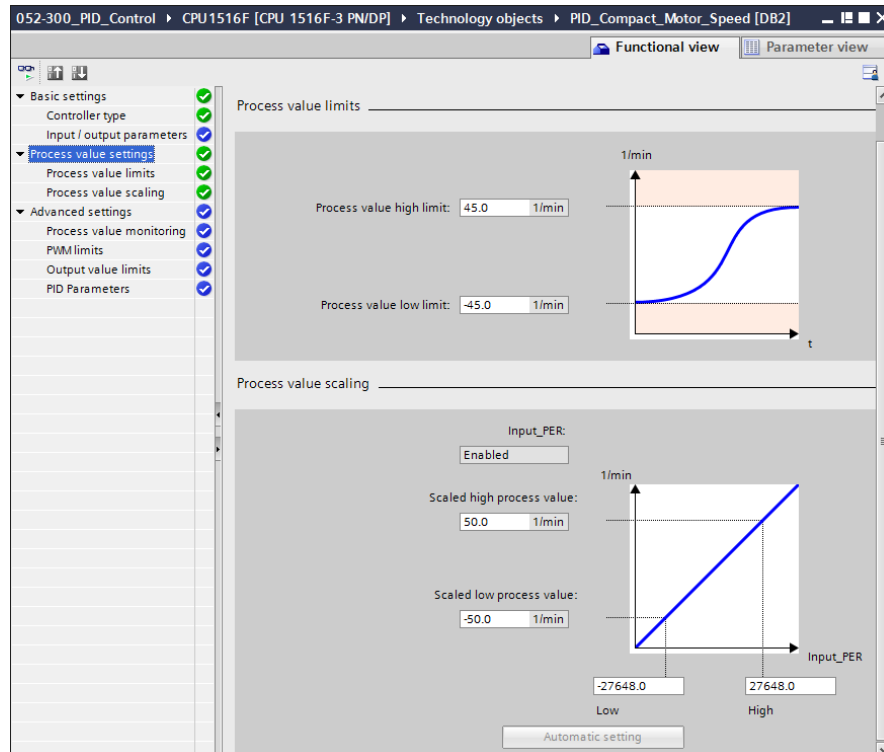
→ In the 'Basic settings', the 'Controller type' and the interconnection of the 'Input / output parameters' are entered. Set the values as shown here.

(→ Basic settings → Controller type → Input / output parameters)



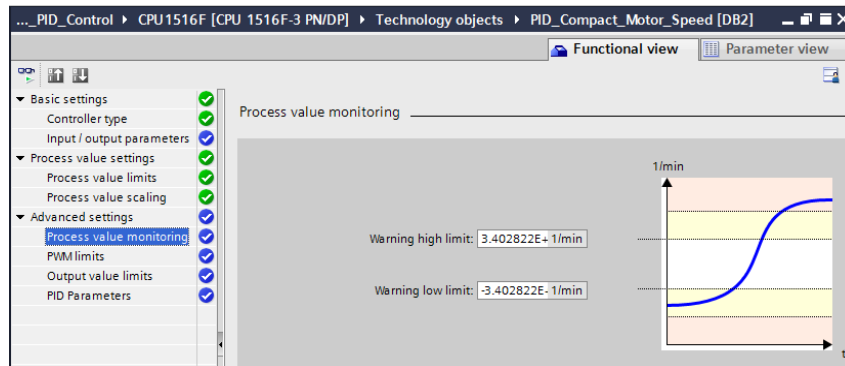
→ In 'Process value settings' we scale to the range +/- 50 rpm and define the 'Process value limits' of +/- 45 rpm.

(→ Process value settings → Process value limits → Process value scaling)



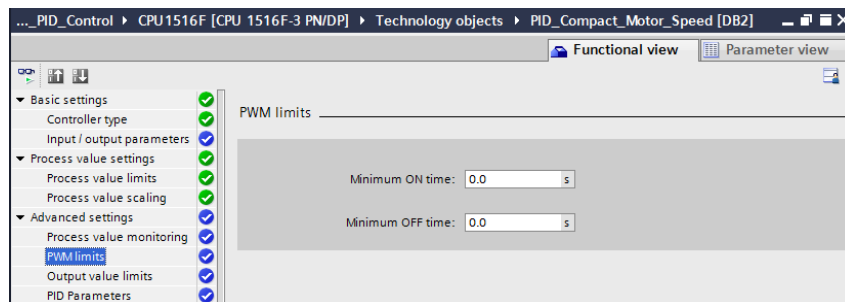
→ In the 'Advanced settings', a process value monitoring would be possible but we don't want to deal with that here.

(→ Advanced settings → Process value monitoring)



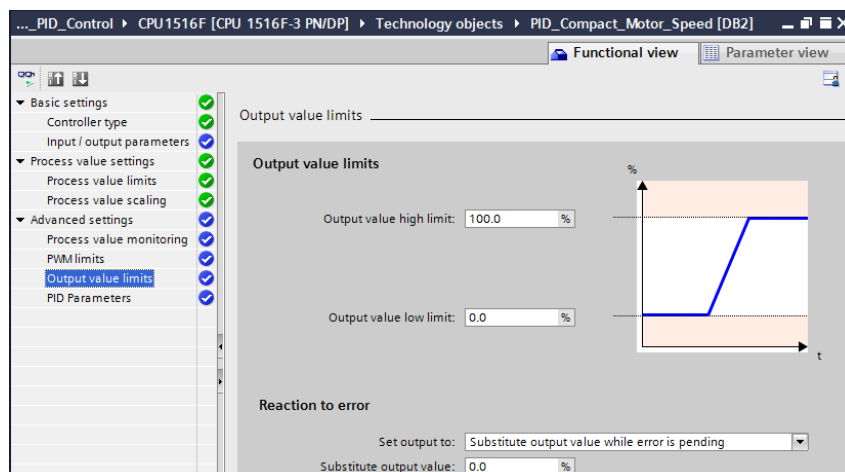
→ In the 'Advanced settings' for 'PWM' (pulse width modulation), we will leave the default values since the output for this is not needed in our project.


(→ Advanced settings → PWM)



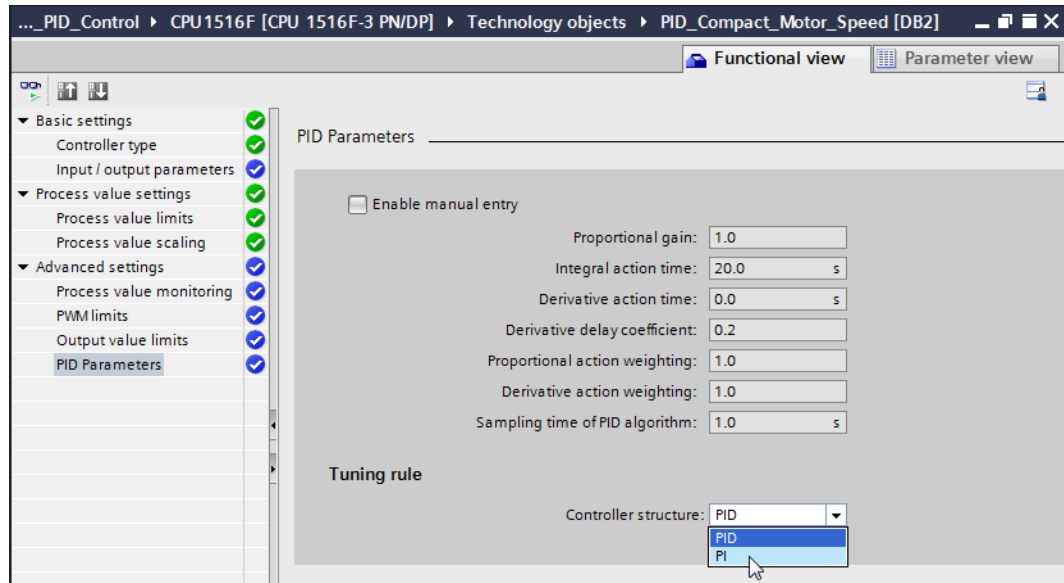
→ In the 'Advanced settings', we define the 'Output value limits' of 0.0 % to 100.0 %.

(→ Advanced settings → Output value limits)





→ In the 'Advanced settings', you will now also find a manual setting of the 'PID parameters'.
Once we have changed the controller structure to 'PI', the configuration window is closed by clicking  and we receive a finished product with a functional PID controller. This should, however, still be commissioned and tuned online during operation.

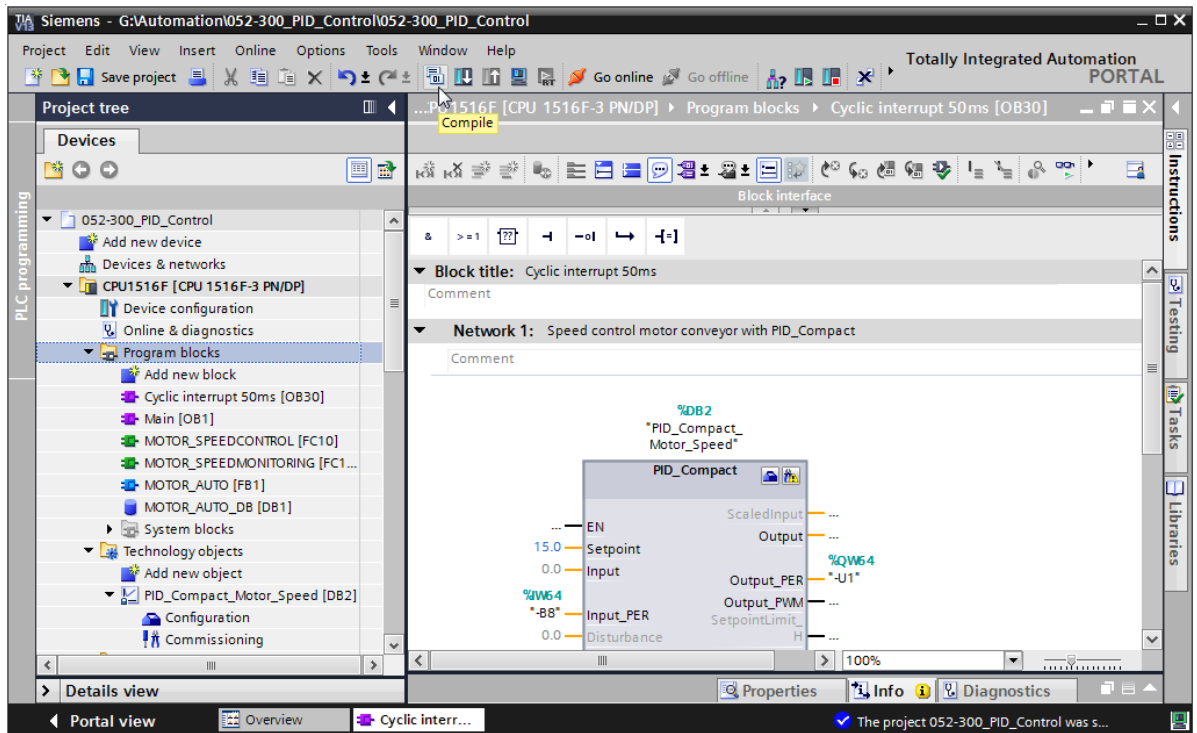
(→ Advanced settings → PID Parameters → Controller structure: PI → )



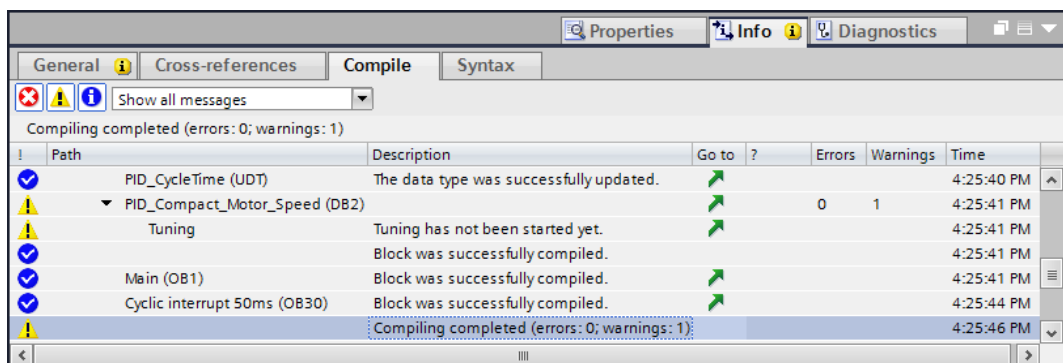
6.3 Save and compile the program

→ To save your project, click the  **Save project** button in the menu. To compile all blocks, click the "Program blocks" folder and select the  icon for compiling in the menu.


(→  **Save project** → Program blocks → )

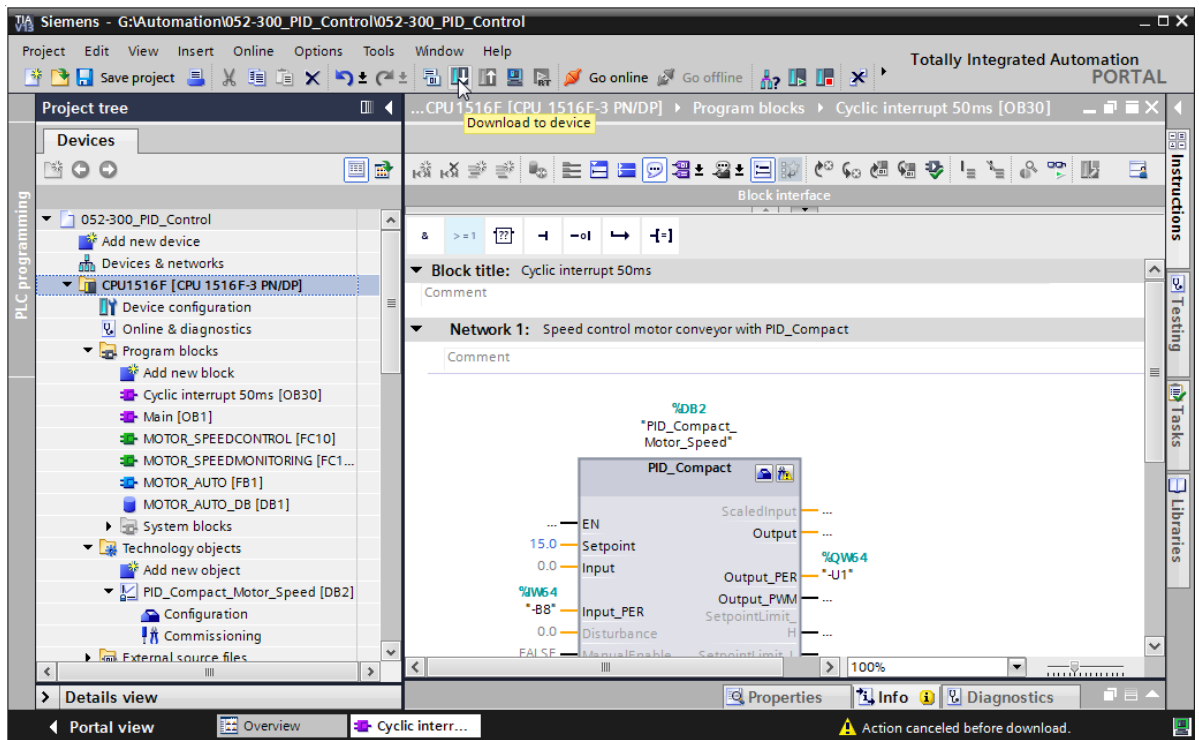


→ The "Info", "Compile" area shows which blocks were successfully compiled.







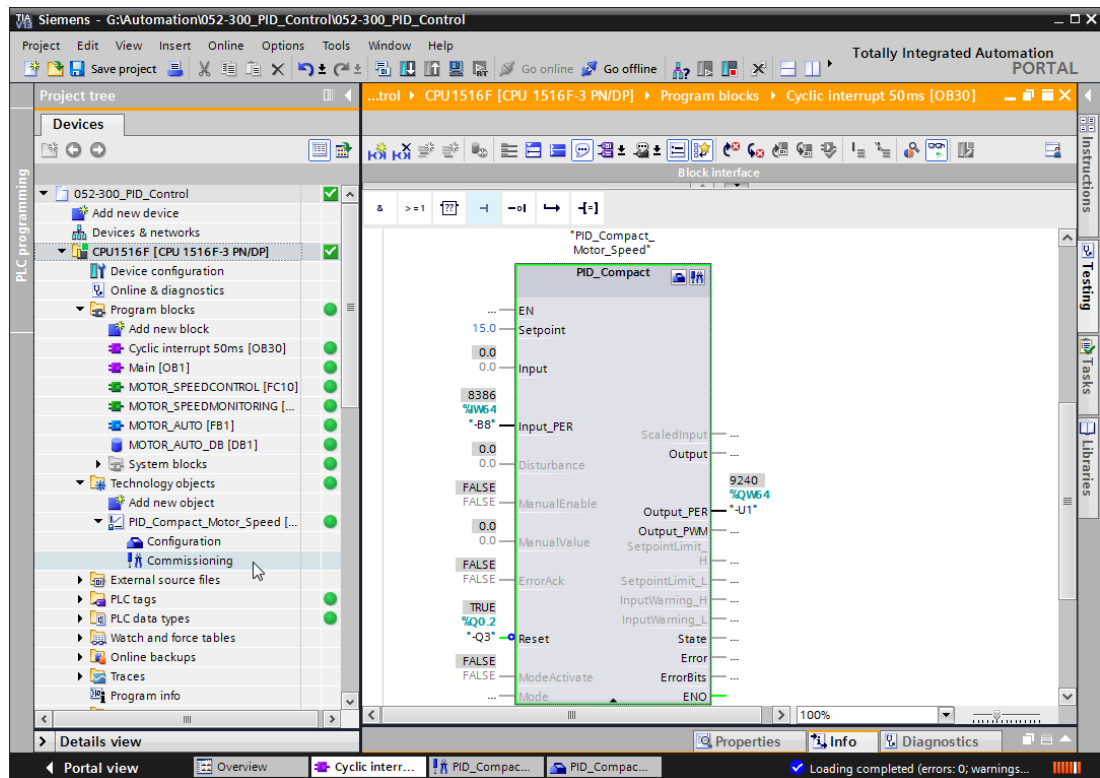
6.4 Download the program

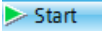
→ After successful compilation, the complete controller with the created program including the hardware configuration can, as described in the previous modules, be downloaded. (→ )

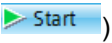


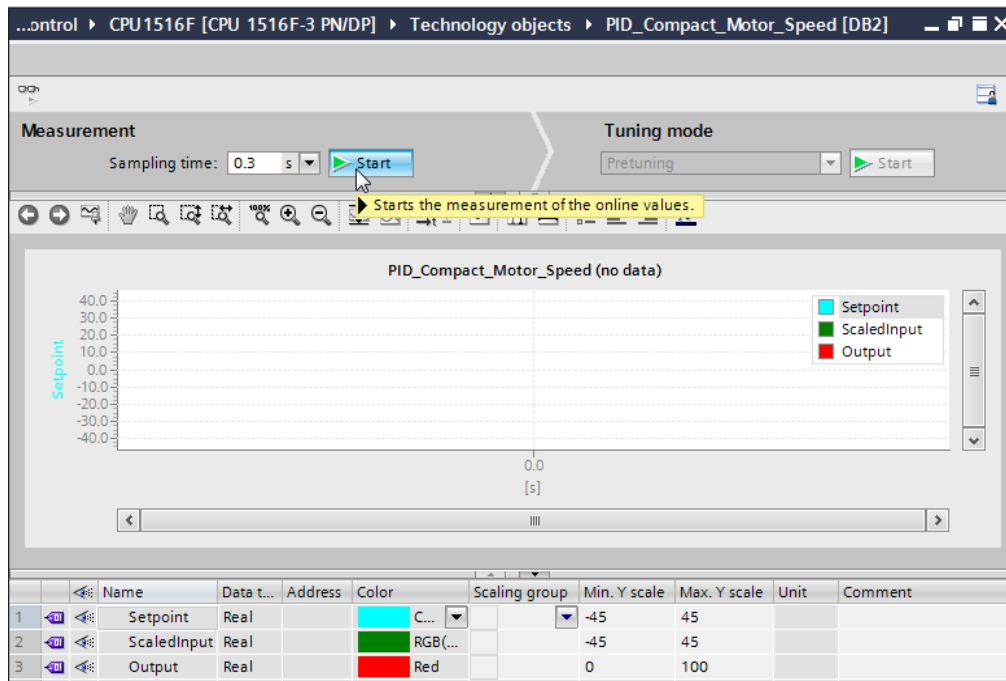
6.5 Monitor PID_Compact

- Click the Monitoring on/off icon  to monitor the state of the blocks and tags when testing the program. At the first start of the CPU, however, the 'PID_Compact' controller is not yet tuned. We still have to start the tuning by clicking the  'Commissioning' icon.
- (→ Cyclic interrupt 50ms [OB30] →  → PID_Compact → )

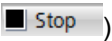


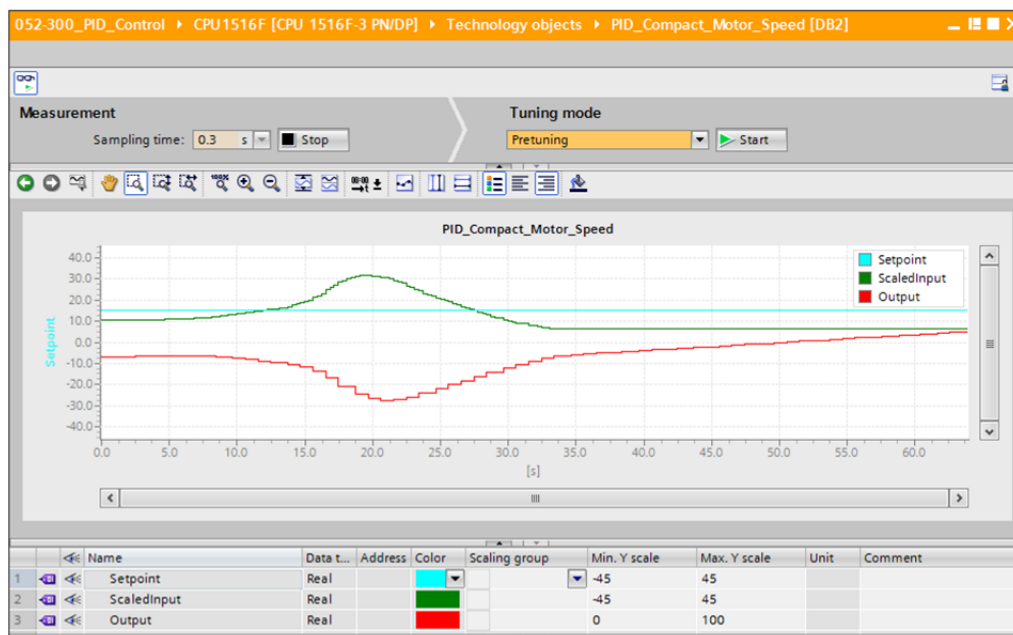
→ If we click  under 'Measurement', the values of the setpoint (Setpoint), actual value (ScaledInput) and manipulated variable (Output) can be displayed and monitored in a diagram.

(→ )



→ The measurement can be stopped again by clicking .

(→ )



6.6 PID_Compact pretuning

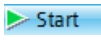
The pretuning determines the process response to a step change of the output value and searches for the turning point. The PID parameters are calculated from the maximum slope and the dead time of the controlled system. The optimal PID parameters are obtained when you perform pretuning and fine tuning.

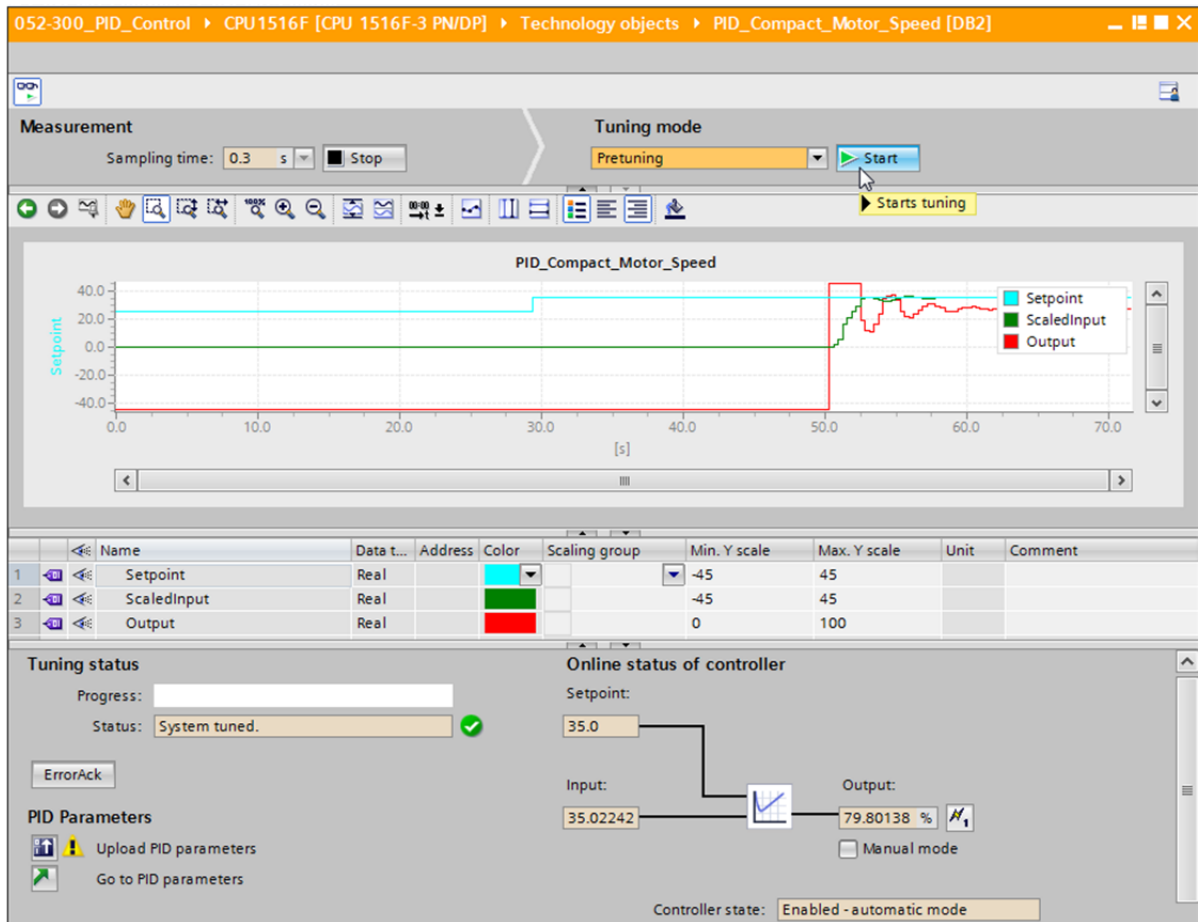
The more stable the actual value is, the easier and more accurately the PID parameters can be determined. Actual value noise is acceptable as long as the actual value rise is significantly greater than the noise. This is most likely the case in "Inactive" or "Manual mode" operating mode. The PID parameters are backed up before they are recalculated.

The following requirements must be met:

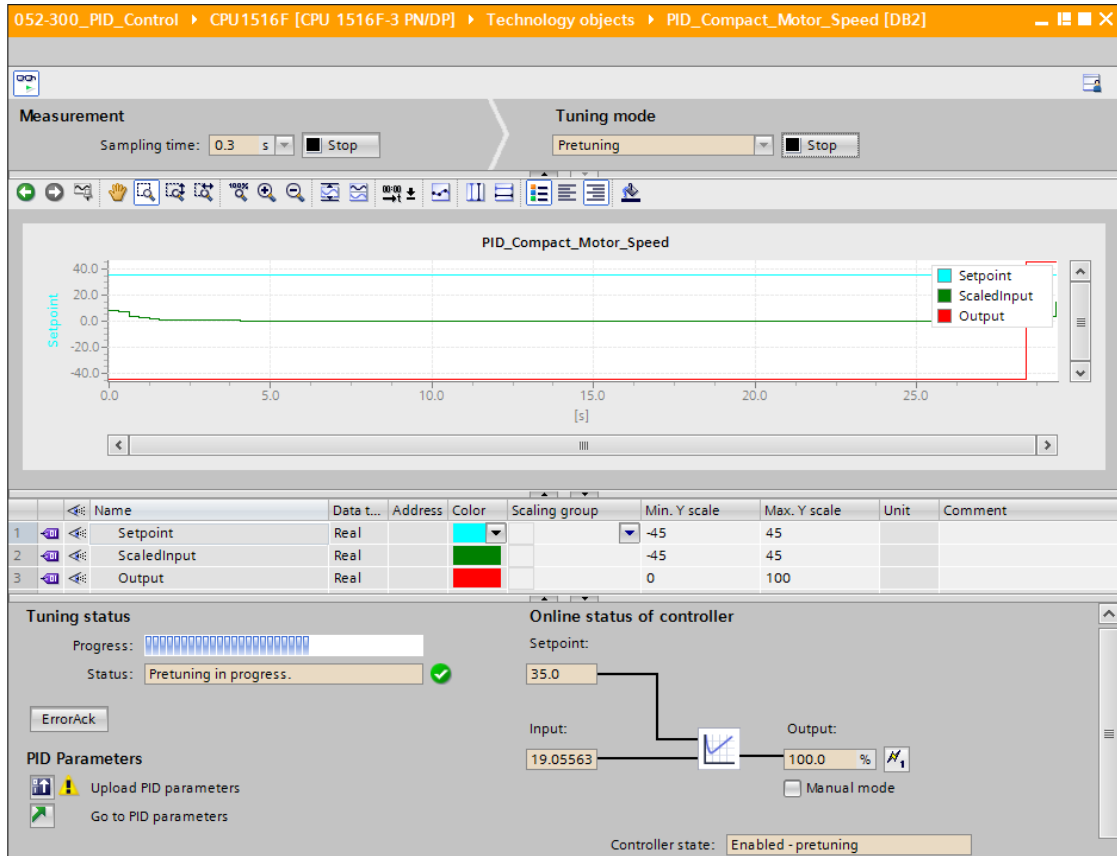
- The "PID_Compact" instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- Reset = FALSE
- PID_Compact is in "Manual mode", "Inactive" or "Automatic mode" operating mode.
- The setpoint and actual value are within the configured limits (see "Process value monitoring" configuration).
- The difference between setpoint and actual value is greater than 30 % of the difference between the process value high limit and low limit.
- The difference between setpoint and actual value is > 50 % of the setpoint.

→ 'Pretuning' is selected as the 'Tuning mode' and this is then started.

(→ Tuning mode → Pretuning → )



→ The pretuning starts. The current work steps and any errors that occur are shown in the "Tuning status" field. The progress bar shows the progress of the current work step.



6.7 PID_Compact fine tuning

The fine tuning generates a constant, limited oscillation of the actual value. The PID parameters are optimized for the operating point based on the amplitude and frequency of this oscillation. All PID parameters are recalculated from the results. The PID parameters resulting from fine tuning generally produce a better response to setpoint changes and disturbances than the PID parameters from pretuning. The optimal PID parameters are obtained when you perform pretuning and fine tuning.

PID_Compact automatically attempts to generate an oscillation that is greater than the actual value noise. The fine tuning is influenced only slightly by the stability of the actual value. The PID parameters are backed up before they are recalculated.

The following requirements must be met:

- The "PID_Compact" instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- Reset = FALSE
- The setpoint and actual value are within the configured limits.
- The control loop is stable at the operating point. The operating point is reached when the actual value is equal to the setpoint.
- No disturbances are expected.
- PID_Compact is in "Manual mode", "Inactive" or "Automatic mode" operating mode.

The fine tuning runs as follows when started in automatic mode:

When you want to improve the existing PID parameters by tuning them, start the fine tuning from automatic mode.

PID_Compact uses the existing PID parameters for controlling until the control loop is stable and the requirements for fine tuning are met. Only then does the fine tuning start.

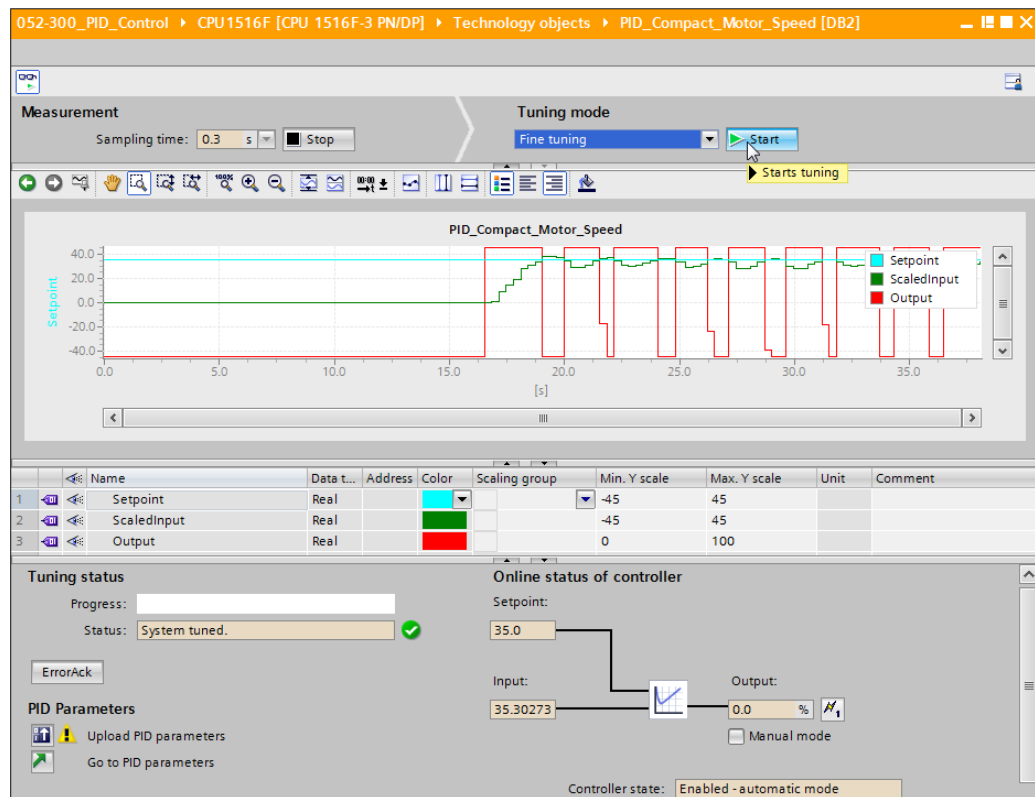
The fine tuning runs as follows when started in inactive or manual mode:


When the requirements for pretuning are met, pretuning is started. PID_Compact uses the determined PID parameters for controlling until the control loop is stable and the requirements for fine tuning are met. Only then does the fine tuning start. If pretuning is not possible, PID_Compact responds as configured in Response to error.

If the actual value is already too close to the setpoint for pretuning, an attempt is made to reach the setpoint with minimum or maximum output value. This can cause increased overshoot.

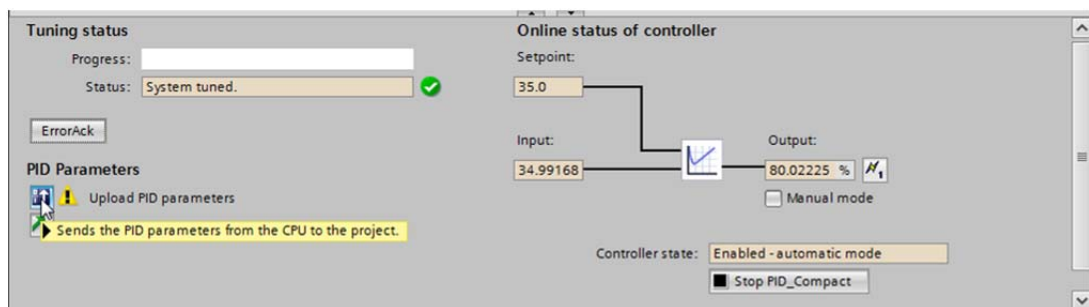
→ 'Fine tuning' is selected as the 'Tuning mode' and this is then started.


(→ Tuning mode → Fine tuning → )




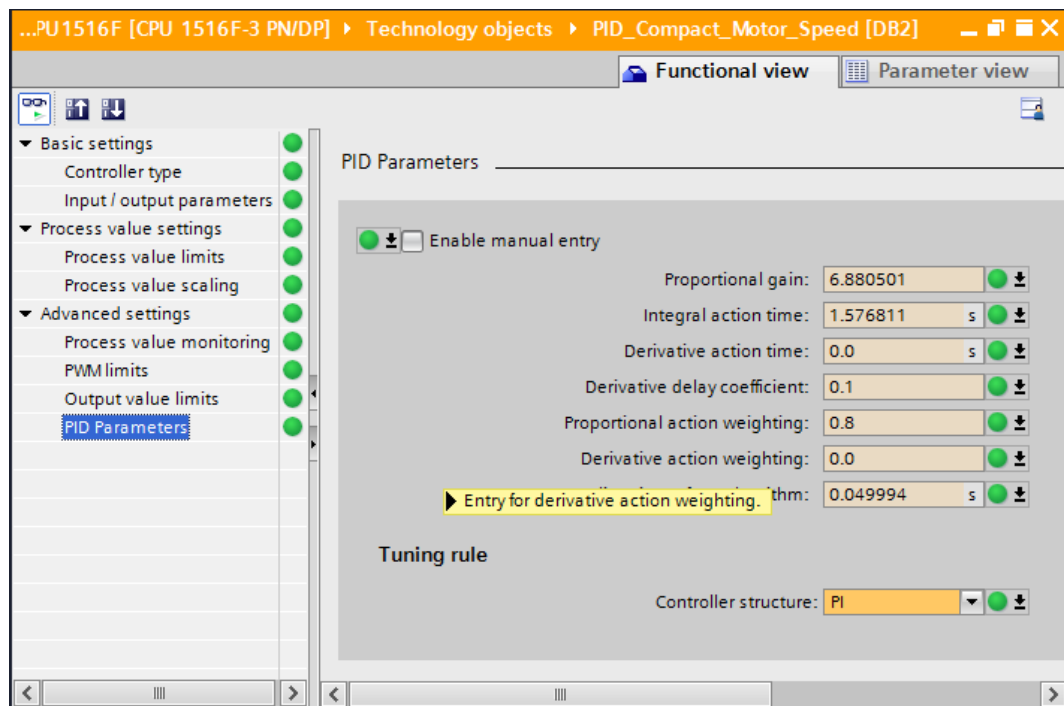
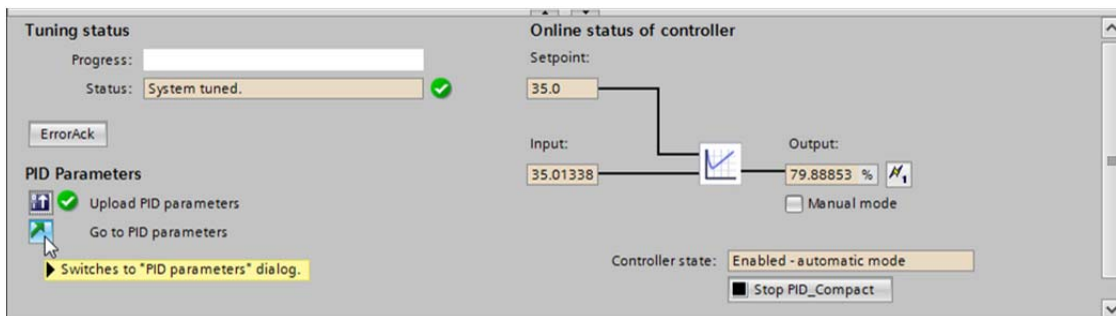
→ The fine tuning starts. The current work steps and any errors that occur are shown in the "Tuning status" field. If the self-tuning was completed without error message, the PID parameters have been tuned. The PID controller switches to automatic mode and uses the tuned parameters. The tuned PID parameters are retained at a Power ON and restart of the CPU. You can download the PID parameters from the CPU to your project with the  button.

(→ )





→ The PID parameters in the configuration can be displayed by clicking .

(→ )



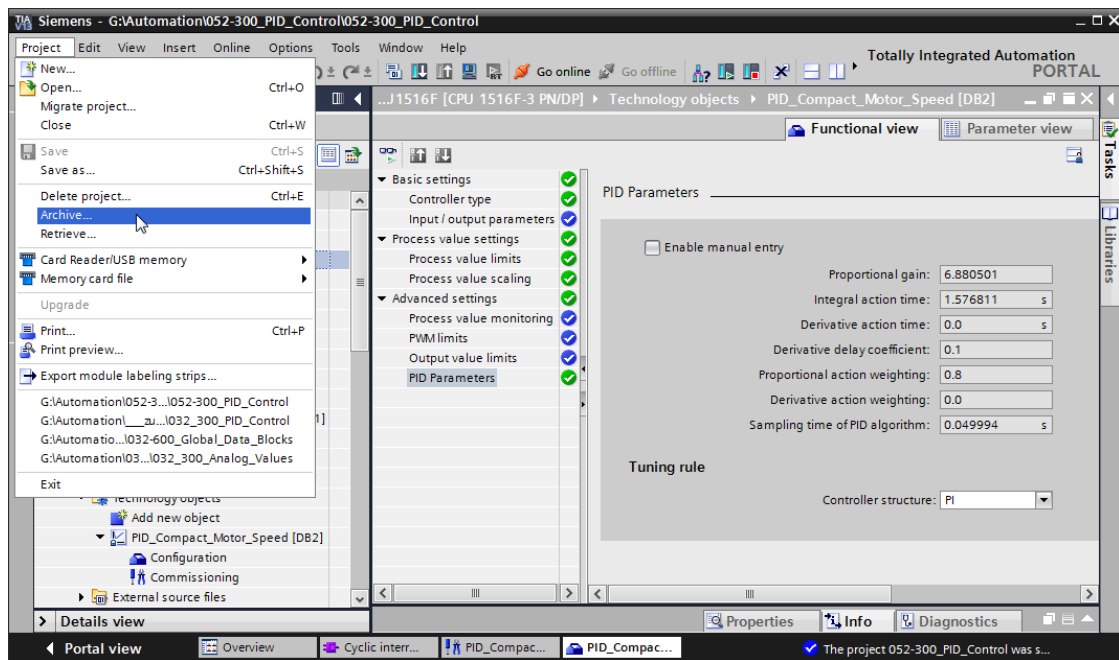
→ As the final step, the online connection should be disconnected and the complete project should be saved.

(→  Go offline →  Save project)

6.8 Archive the project

→ Now we want to archive the complete project. Select the → 'Archive ...' command in the → 'Project' menu. Select a folder where you want to archive your project and save it with the file type "TIA Portal project archive".

(→ Project → Archive → TIA Portal project archive → 052-300_PID_Controller.... → Save)



7 Checklist

No.	Description	Completed
1	Cyclic interrupt OB Cyclic interrupt 50ms [OB30] successfully created.	
2	PID_Compact controller in cyclic interrupt OB Cyclic interrupt 50ms [OB30] called and connected.	
3	Configuration of the PID_Compact controller performed.	
4	Compiling successful and without error message	
5	Download successful and without error message	
6	Pretuning successful and without error message	
7	Fine tuning successful and without error message	
8	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor part at slide activated (-B4 = 1) then Conveyor motor -M1 variable speed (-Q3 = 1) switches on and stays on. The speed corresponds to the speed setpoint in the range +/- 50 rpm	
9	Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds)	
10	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0	
11	Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0	
12	Manual mode (-S0 = 0) → -Q3 = 0	
13	Switch off station (-K0 = 0) → -Q3 = 0	
14	Cylinder not retracted (-B1 = 0) → -Q3 = 0	
15	Speed > Motor_speed_monitoring_error_max → -Q3 = 0	
16	Speed < Motor_speed_monitoring_error_min → -Q3 = 0	
17	Project successfully archived	

8 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

www.siemens.com/sce/s7-1500

[illegible]

SIEMENS

Simplify your education in automation

Unique support for educators and students in educational institutions

Siemens Automation Cooperates with Education

Siemens Automation Cooperates with Education (SCE) focuses on the needs of students by providing educators with the tools and training to simplify their education in automation. The SCE program delivers value by offering specific courses, training curriculums and exceptional Hardware and Software Trainer Packages.

We are committed to sharing knowledge, resources and tools for teaching automation and drive system technologies through partnerships with educational institutions.

SIEMENS

Global Industry
Partner of
WorldSkills
International



siemens.com/sce

Get more information

Siemens Automation Cooperates with Education:
www.siemens.com/sce

SCE training curriculums:
www.siemens.com/sce/curriculums

SCE trainer packages:
www.siemens.com/sce/tp

SCE trial software/firmware:
www.siemens.com/sce/trial

SCE contact partners:
www.siemens.com/sce/contact

SCE Support Finder:
www.siemens.com/sce/supportfinder

SIMATIC Controller:
www.siemens.com/controller

Distributed IO:
www.siemens.com/et200

Totally Integrated Automation:
www.siemens.com/tia

Totally Integrated Automation Portal:
www.siemens.com/tia-portal

SIMATIC Manual Guide:
www.siemens.com/simatic-docu

Information material for SIMATIC:
www.siemens.com/simatic/printmaterial

Siemens Industry Online Support – Technical Forum:
www.automation.siemens.com/WW/forum

Service&Support:
www.siemens.com/automation/support

Industry Mall for electronic ordering:
www.siemens.com/industrymall

Industrial Security

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, solutions, machines, equipment and/or networks. They are important components in a holistic industrial security concept. With this in mind, Siemens' products and solutions undergo continuous development. Siemens recommends strongly that you regularly check for product updates.

For the secure operation of Siemens products and solutions, it is necessary to take suitable preventive action (e.g. cell protection concept) and integrate each component into a holistic, state-of-the-art industrial security concept. Third-party products that may be in use should also be considered. For more information about industrial security, visit
<http://www.siemens.com/industrialsecurity>.

To stay informed about product updates as they occur, sign up for a product-specific newsletter. For more information, visit
<http://support.automation.siemens.com>.

SIEMENS

Global Industry
Partner of
WorldSkills
International



Siemens AG
Digital Factory Division
P.O. Box 48 48
90026 Nuremberg
GERMANY

Subject to change without prior notice
Article No. BTCG-A10001-01-7600
B.2112.W2.SCE.014 / Dispo 06321
SO 0416 1. DPG 532 En
Printed in Germany
© Siemens AG 2016

The information provided in this brochure contains merely general descriptions or characteristics of performance which in case of actual use do not always apply as described or which may change as a result of further development of the products. An obligation to provide the respective characteristics shall only exist if expressly agreed in the terms of contract. Availability and technical specifications are subject to change without notice.
All product designations may be trademarks or product names of Siemens AG or supplier companies whose use by third parties for their own purposes could violate the rights of the owners.